Attacks on quantum key distribution protocols that employ non-ITS authentication

Christoph Pacher[1], Aysajan Abidin[2], Thomas Lorünser[1], Momtchil Peev[1], Rupert Ursin[3], Anton Zeilinger[3], Jan-Åke Larsson[2]

[1] Department of Safety & Security, AIT Austrian Institute of Technology, Austria
[2] Department of Electrical Engineering, Linköping University, Linköping, Sweden
[3]Institut für Experimentalphysik, Universität Wien, Austria and Institute for Quantum Optics and Quantum Information, Austrian Academy of Sciences, Austria

E-mail: Christoph.Pacher@ait.ac.at

Abstract.

We demonstrate how adversaries with unbounded computing resources can break Quantum Key Distribution (QKD) protocols which employ a particular message authentication code suggested previously. This authentication code, featuring low key consumption, is not Information-Theoretically Secure (ITS) since for each message the eavesdropper has intercepted she is able to send a different message from a set of messages that she can calculate by finding collisions of a cryptographic hash function. However, when this authentication code was introduced it was shown to prevent straightforward Man-In-The-Middle (MITM) attacks against QKD protocols.

In this paper, we prove that the set of messages that collide with any given message under this authentication code contains with high probability a message that has small Hamming distance to any other given message. Based on this fact we present extended MITM attacks against different versions of BB84 QKD protocols using the addressed authentication code; for three protocols we describe every single action taken by the adversary. For all protocols the adversary can obtain complete knowledge of the key, and for most protocols her success probability in doing so approaches unity.

Since the attacks work against all authentication methods which allow to calculate colliding messages, the underlying building blocks of the presented attacks expose the potential pitfalls arising as a consequence of non-ITS authentication in QKD-postprocessing. We propose countermeasures, increasing the eavesdroppers demand for computational power, and also prove necessary and sufficient conditions for upgrading the discussed authentication code to the ITS level.

1. Introduction

Quantum key distribution (QKD) is a cryptographic key-agreement protocol consisting of two steps: quantum communication and measurements, and classical post processing. The outstanding property of QKD is that it is an Information-Theoretically Secure (ITS) and universally composable (UC) protocol given that its classical communication is performed over an authentic channel (note that *all* key-agreement protocols are insecure over non-authentic channels). Thus QKD is a very powerful cryptographic primitive but in order to be useful for practical key agreement purposes it must be *composed* with an independent primitive enforcing the mentioned requirement for authenticity of classical communication.

The standard cryptographic approach ensuring authenticity of communication messages against malicious attackers is to use a message authentication code (MAC) [1]. A convenient class of MACs are *systematic* MACs which replace the original message with a concatenation of the message itself and an additional tag which is the image of a keyed hash function applied to the message. It is well-known that Strongly Universal$_2$ (SU$_2$), and more generally Almost Strongly Universal$_2$ (ASU$_2$) hashing (see Appendix C) is an ITS primitive that can be used to calculate systematic MAC tags.

1.1. Related work

Very recently authentication based on ASU$_2$ hashing was explicitly shown [2] to be also UC (a fact that has been used implicitly for quite some time). Therefore UC message authentication with ASU$_2$ hashing can be composed with UC quantum key distribution over authentic channels to form a UC (quantum-classical) key agreement protocol over non-authentic channels. Thus, ASU$_2$ hashing is sufficient for the authentication of the classical messages exchanged during any QKD protocol. However, although composing two UC primitives is *sufficient* for getting a UC composed protocol this is not a priori *necessary* as in principle it is not excluded that it can be shown directly that the final protocol is UC. In this sense it might still be possible that QKD over non-authentic channels can be made secure without relying on ASU$_2$ hashing. Alternatives using weaker authentication have been proposed, and this paper focuses on the method of Ref. [3], that puts forward a hash function which is a composition of an (inner) known public hash function (like SHA) and an (outer) SU$_2$ function. It was proven that QKD using this authentication is secure against an eavesdropper that attempts to break the protocol using a straightforward "man-in-the-middle" (MITM) attack, as defined below. Later, in Refs. [4, 5] it was observed that an eavesdropper can apply more advanced strategies than a straightforward MITM and get a significant leverage by being able to break QKD with particular realizations of post-processing. It has, however, been argued [3, 6] that this weakness occurs only in specific post processing realizations, while in practical (or generic) ones the proposed eavesdropping techniques remain inadequate.

1.2. New results

In this paper we use the adversarial approaches of [4, 5], extend them significantly to full scale eavesdropping strategies, and demonstrate in detail how to break several explicit QKD protocols, that employ the authentication method of Ref. [3], under the assumptions that the adversary possesses unbounded computation resources and in some cases quantum memory. The general attack-pattern is a sophisticated (interleaving) MITM attack, in which the adversary (Eve) carries out independent protocols with the legitimate parties (Alice and Bob). In doing so Eve manages to modify her respective protocol messages such that these collide with those of Alice and Bob under the first part of the authentication of Ref. [3]. Depending on the protocol variants (e.g., immediate vs. delayed authentication), the different attacks which we study address sifting, error correction, confirmation, and privacy amplification or only some of these steps. These techniques can be used to break a very broad class of post-processing protocol realizations which include those routinely used in practical implementations. With significant probability that in most attacks approaches unity Eve shares a key with the legitimate parties.

We also consider some countermeasures, which consist of modifications of the two-step authentication mechanism. These modifications result in a range of complications to Eve: (i) increasing Eve's computational load substantially, (ii) forcing her to do considerable online computation rather than offline; and (iii) depriving her of any attack potential by finally re-establishing ITS for the modified construction. We give necessary and sufficient conditions for ITS with this construction; that the conditions are sufficient is already known from earlier results, but that the conditions are necessary is, as far as we know, a new result.

1.3. Structure of the paper

Section 2 contains a motivation on why authentication is needed in QKD, shortly reviews message authentication codes and Universal hashing, and gives a more detailed description of the authentication method under study here. Section 3 introduces the attack vectors and then details three different QKD protocols and attacks against them in a step-by-step fashion. In Tables 2 and 3 we summarize the attacks and the gained knowledge on the key for each of them, as well as for a number of further protocol versions. Section 4 discusses how the security of the authentication method can be improved and presents a theorem that gives necessary and sufficient conditions for ITS of the modified method. The conclusions and outlook are given in Section 5. The Appendices contain technical proofs and summarize some definitions of Universal hash function families.

2. Authentication in QKD

The need for authentication becomes clear if we consider for a moment the opposite case, i.e. an "unprotected" channel that allows arbitrary modification of messages in transit.

2.1. Man-in-the-middle attacks and Message Authentication Codes

The unprotected channel will enable a straightforward "man-in-the-middle" (MITM) attack:

Definition 1 (straightforward man-in-the-middle (MITM) attack)**.** *In the* straightforward man-in-the-middle attack *the eavesdropper (Eve) builds or buys a pair of QKD devices identical to those of the legitimate parties (Alice and Bob) and cuts "in the middle" the quantum and classical communication channels connecting Alice and Bob. She now connects each of her devices to the "loose ends" of the quantum and classical channels and launches two* independent *QKD sessions, one with Alice and the other with Bob. Eve effectively pretends to be Bob to Alice and Alice to Bob. Eventually she shares a (different) key with each of the legitimate parties which allows her to communicate with them independently. If Alice sends an encrypted message to Bob, Eve can intercept the message and decrypt it, encrypt it with the key she shares with Bob, and send it to Bob.*

Alice and Bob never come to realize that the security of their communication is completely lost. This is completely analogous to the classic MITM attack against the unauthenticated Diffie-Hellman key agreement protocol [1, Chap. 12.9.1]. Obviously, any (classic or quantum) key agreement protocol that has no proper authentication (or integrity check) of messages exchanged between the communicating parties can be broken with a similar impersonation attack.

So ideally an adversary should not be able to insert messages into the channel, and moreover messages sent by one legitimate user to the other are always delivered and are not modified. However, there are no a-priori authentic communication channels. Appending a so-called Message Authentication Code (MAC) to each communication message can mimic an authentic channel, but cannot guarantee delivery of messages, as these can in practice always be blocked.

Definition 2 (Message Authentication Code (MAC)[1])**.** *A* Message Authentication Code (MAC) algorithm *is a family of functions h_K parameterized by a secret key K with the following properties: (i) given a message x and a key K, the MAC value $h_K(x)$ (also called* tag*) should be easy to compute, (ii) it maps a message x of arbitrary finite bitlength to a tag $h_K(x)$ of fixed bitlength n, and (iii) given a description of the function family h, for every fixed allowable value of K it must be computation-resistant. The last property means that given zero or more message-tag pairs $(x_i, h_K(x_i))$ it is computationally infeasible to compute any message-tag pair $(x, h_K(x))$ for any new input $x \neq x_i$ without knowing K.*

Normally, MACs are either based on (a) cryptographic hash functions (e.g. HMAC-SHA-256 based on SHA-256), on (b) block cipher algorithms (e.g. AES-CMAC based on AES), or on (c) Universal$_2$ hashing (see Appendix C). Message authentication codes based on (a) or (b) typically use one key for many messages, and offer computational security, i.e. they can only be broken with sufficient computing power (or when a hidden weakness of the algorithm is detected).

2.2. Universal hashing and UC security

MACs based on Universal$_2$ hashing have to use one (new) key per message, but offer information-theoretic security which is independent of the adversary's computing power. In more detail, for SU$_2$ hash functions, a random guess of the MAC tag is provably the best possible attack, while ϵ-ASU$_2$ hash functions still provide a strict upper bound (namely ϵ) on the attacker's success probability to substitute an observed message-tag pair with another valid message-tag pair (*substitution attack*) or to insert a valid message-tag pair.

Universal hashing was originally proposed by Wegman and Carter [7, 8]. It was identified as an appropriate match for QKD, as Wegman-Carter's and later constructions [9–12] consume relatively low amount of key. The aim is to have less key consumption than the key generation in a typical QKD session [13], so that each session can reserve a portion of its output for authentication of the subsequent one. Then, the process only needs to be kick-started by an initial, one-time, pre-distributed secret.

Security analysis of QKD (see, e.g., Ref. [14] and references therein for a recent overview) has typically been based on the requirement that the classical post-processing communication is secured by a MAC based on Universal hashing, to upper bound an adversary's chances to modify or insert messages without getting detected. In addition UC-security definitions for QKD have been established [15–18]. As a consequence combining the two ε-UC-secure protocols QKD and ASU$_2$ authentication yields a joint, *UC-secure key growing mechanism* over non-authentic classical channels (see [2]). Thus, MACs based on ASU$_2$ hashing are sufficient for security, but it is an open question whether they are also necessary, and what security would be obtained for other alternatives.

2.3. The non-ITS authentication mechanism of Ref. [3]

The authentication mechanism proposed in Ref. [3] aimed to consume less key than ASU$_2$ authentication. The intended goal is a positive key balance of the combination QKD plus authentication even in realizations that use (relatively) short blocks in the post processing step. Note that later experimental progress has made these objectives not so relevant, as short key blocks are no longer necessary from an implementation perspective [19]. Still, a complete security analysis of the authentication mechanism of [3] is intriguing from a theoretical point of view as the mechanism has interesting properties not shared by any of the methods mentioned above.

To start with, we summarize the proposal of Ref. [3] and introduce some notation (see also Table 1). The proposal relies on a two-step hash function evaluation: $t = g_K(m) := h_K(f(m))$, where $f : \mathcal{M} \to \mathcal{Z}$ is a publicly known hash function and $h_K : \mathcal{Z} \to \mathcal{T}$ belongs to an SU_2 hash function family $\mathcal{H}$ (see Appendix C). Here, $\mathcal{M}$ is the set of messages to be authenticated, $\mathcal{Z}$ is an intermediate set of strings, and $\mathcal{T}$ is the set of tags with $|\mathcal{M}| \gg |\mathcal{Z}| > |\mathcal{T}|$.

2.3.1. Insertion of messages is ruled out Now assume that Eve attempts to calculate or guess the tag for a fixed message $m^{\mathbf{E}}$ that she wants to insert. In that case she has a success probability of $1/|\mathcal{T}|$ (irrespective of her computing power). This is because the key K which identifies the SU_2 hash function is not known to her. Thus, the authentication mechanism is (first-)preimage resistant, i.e. knowledge of the authentication tag alone does not allow to find messages yielding the same tag.

2.3.2. Substitution with given messages is ruled out Let us further assume, Eve has intercepted a (valid) message-tag pair $(m^{\mathbf{A}}, t)$ from Alice and wants to substitute it with her *fixed* message $m^{\mathbf{E}}$ and some tag. Then Eve's chances increase slightly because she now has access to the intermediate value $f(m^{\mathbf{A}})$, and can check if $f(m^{\mathbf{A}}) = f(m^{\mathbf{E}})$. If there is a collision, Eve knows that $(m^{\mathbf{E}}, t)$ is a valid message-tag pair and can just send this, otherwise she guesses the tag as above. The total probability of success is now bounded by the guessing probability plus the collision probability, and assuming that $m^{\mathbf{A}}$ is random to Eve and that f is a good hash function, the collision probability is low (for details see [3]). So this two-step authentication works well in a situation when Eve is given a fixed message $m^{\mathbf{E}}$ to generate the tag for. One immediate consequence is that Eve cannot perform the straightforward MITM attack (cf. Definition 1) with significant success probability since she would need to generate tags for messages $m^{\mathbf{E}}$ from her devices without knowledge of K, for which case the above bound applies.

2.3.3. The weakness However, one should note that using the intercepted message-tag pair $(m^{\mathbf{A}}, t)$ and enough computational power, Eve can in principle search for other preimages of t under f. If she can find (at least) one message $\tilde{m}^{\mathbf{E}}$ such that $f(m^{\mathbf{A}}) = f(\tilde{m}^{\mathbf{E}})$ then $h_K(f(m^{\mathbf{A}})) = h_K(f(\tilde{m}^{\mathbf{E}}))$ and therefore $(\tilde{m}^{\mathbf{E}}, t)$ is a valid message-tag pair *for any key* K. She can now replace $m^{\mathbf{A}}$ with $\tilde{m}^{\mathbf{E}}$ with success probability of 100%. The question now is if this (one of these) $\tilde{m}^{\mathbf{E}}$ can be used in place of the message $m^{\mathbf{E}}$. It would seem that, if Eve strictly follows the appropriate QKD protocol (random settings, best possible bit error rate, ...), this is not possible.

However, Eve is not forced to follow the precise requirements of the QKD protocol [5]; she only needs to make it seem to Alice and Bob that she does so. For example, Eve does not need to use random settings (e.g. preparation bases and raw keys), or even correctly send all settings she used. If it helps her, she can use a fixed sequence of settings or report other settings for some qubits than the ones actually used.

An early suggestion [4] was to *select* the privacy amplification map carefully, rather than generating it randomly. This would give Eve a shared key with Bob, but not with Alice. Later, as mentioned above, it was observed that Eve may deviate from the QKD protocol in several places [5]. If Eve uses a fixed sequence of settings (e.g. measurement and preparation bases) on the quantum channel this would enable her to do the calculations for finding $\tilde{m}^{\mathbf{E}}$ offline. If Eve sends the wrong settings for some of the qubits this will allow her to choose from several $\tilde{m}^{\mathbf{E}}$, to get a collision. This would constitute the basis for a sophisticated MITM attack that can break simplified QKD protocols. In these simplified protocols, the breaches could be closed by relatively straightforward countermeasures [6], but the security of the standard and/or hardened protocols remained an open issue. We aim to settle this in the present paper.

3. Attacks against non-ITS authentication in QKD

In this section, we give detailed descriptions of four different attacks on three different explicit QKD protocols. We also give an overview of the effectiveness of this kind of attacks against other QKD protocols, and for different types of resources available to Eve. In each case, the essence of the attack is to intercept a valid message-tag pair (sent by Alice or Bob) and—using large computational resources and/or leveraging weaknesses of the public hash function algorithm—find further preimages of the tag (messages that hash to the same hash value as the intercepted message) that are used by the eavesdropper.

3.1. Hash collisions

Assume that Eve has intercepted a message-tag pair $(m^{\mathbf{A}}, t)$ from Alice. The following lemma states that (under a mild assumption) for any fixed message $m^{\mathbf{E}}$, that Eve would like to send, there exists with probability almost 1 a message $\tilde{m}^{\mathbf{E}}$, such that (i) $\tilde{m}^{\mathbf{E}}$ is almost identical to $m^{\mathbf{E}}$, i.e. $\tilde{m}^{\mathbf{E}}$ has *small Hamming distance* to $m^{\mathbf{E}}$, and (ii) $(\tilde{m}^{\mathbf{E}}, t)$ will be accepted as authentic, i.e. $h_K(f(\tilde{m}^{\mathbf{E}})) = t$.

Lemma 1. *Let $\mathcal{B}$ be the closed ball of all messages m having a Hamming distance to $m^{\boldsymbol{E}}$ not exceeding w:*

$$\mathcal{B} = \left\{ m : d_H(m, m^{\boldsymbol{E}}) \leq w \right\},$$

and let us assume that f maps all messages in $\mathcal{B}$ randomly onto $\mathcal{Z}$. Then the probability that at least one of the messages in $\mathcal{B}$ is validated by the given tag $t = h_K(f(m^{\boldsymbol{A}}))$ is

$$\mathcal{P}_{coll}^{succ} = \Pr\left\{ \exists \tilde{m}^{\boldsymbol{E}} \in \mathcal{B} : h_K(f(\tilde{m}^{\boldsymbol{E}})) = t) \right\} > 1 - \exp\left(-|\mathcal{B}||\mathcal{Z}|^{-1} \right).$$

For simplicity we can loosen the bound and replace $|\mathcal{B}|$ by $\binom{\ell}{w} < |\mathcal{B}|$, where ℓ is the length of the binary message $m^{\boldsymbol{E}}$.

The proof of Lemma 1 is given in Appendix A. Since no assumptions on the computational power of Eve are imposed, she will be able to find with probability

$\mathcal{P}_{\text{coll}}^{\text{succ}}$ such an $\tilde{m}^{\mathbf{E}}$. For typical parameters, e.g. $|\mathcal{Z}| = 2^{256}$, and $\ell = 2^{12}$ (2^{13}, 2^{14}, 2^{15}, 2^{16}, 2^{17}), a Hamming distance $w = 32$ (28, 25, 22, 20, 19) is sufficient to reach a success probability of 99.9%.

3.1.1. Attacking the sifting stage – hiding in the noise Let us assume that during the sifting stage the legitimate parties will exchange messages that contain one bit per preparation/measurement basis (time slot). Let us assume further that Eve can successfully attack the protocol (as discussed below), if she can substitute such a message, say $m^{\mathbf{A}}$, with a sifting message of her choice, say $m^{\mathbf{E}}$. From Lemma 1 it follows that if Eve replaces $m^{\mathbf{A}}$ with $\tilde{m}^{\mathbf{E}}$ instead of $m^{\mathbf{E}}$, she will introduce at this step (at most) an additional error $\epsilon = w/\ell \approx 0.78\%$ (0.34%, 0.15%, 0.067%, 0.031%, 0.014%) (with parameters from above; in the worst case each modified basis bit could result in one flipped sifted key bit). This strategy allows Eve to hide the substitution of sifting messages in the usual noise on the quantum channel, since the following error correction step will also remove these small additional deviations. Obviously, the larger the message length ℓ, the easier Eve's task is.

3.1.2. Correlating the sifted keys of Alice and Bob Assume for the moment that Eve has intercepted the quantum bits from Alice and has saved them into her quantum memory. Assume further that she managed to fool Alice, so that Alice announces her the corresponding preparation bases. Then Eve can measure the quantum bits and get Alice's raw key.

The strongest of the presented attacks is based on the fact that once Eve knows the raw key of Alice, she can by using a modification of Bob's sifting message ensure with high probability that the complete sifted key of Alice will be almost identical to that of Bob (cf. description of Protocol 1 and step (Se") of the attack against it.).

Lemma 2. *Let $d^{\boldsymbol{A}} \in_R \{0,1\}^n$ be the raw key that Alice has used to prepare her quantum bits. Once Eve knows $d^{\boldsymbol{A}}$ she can determine $\lfloor n/2 \rfloor - k$ bits of any fixed sifted key $s^{\boldsymbol{E}}$ that she wants Alice to create with probability*

$$\mathcal{P}_{\textit{sift-attack}}^{\textit{succ}} \geq 1 - \exp\left(-\frac{2k^2}{n}\right) \tag{1}$$

by replacing Bob's sifting message with a message $b^{\boldsymbol{A}=\boldsymbol{E}}$ that she has prepared.

Eve's attack will succeed if a *subsequence* of $s^{\mathbf{E}}$ (derived by deleting some elements without changing the order) of length at least $\lfloor n/2 \rfloor - k$ is also a *subsequence* of $d^{\mathbf{A}}$. The proof and a simple and efficient algorithm to generate $b^{\mathbf{A}=\mathbf{E}}$ is given in Appendix B. Note, that $k = O(\sqrt{n})$ is sufficient for $\mathcal{P}_{\text{sift-attack}}^{\text{succ}} \approx 1$.

3.2. General remarks, protocol notation and settings used

Any successful attack is based on finding protocol modifications yielding communication messages that collide with those of the legitimate parties under the fixed hash function in

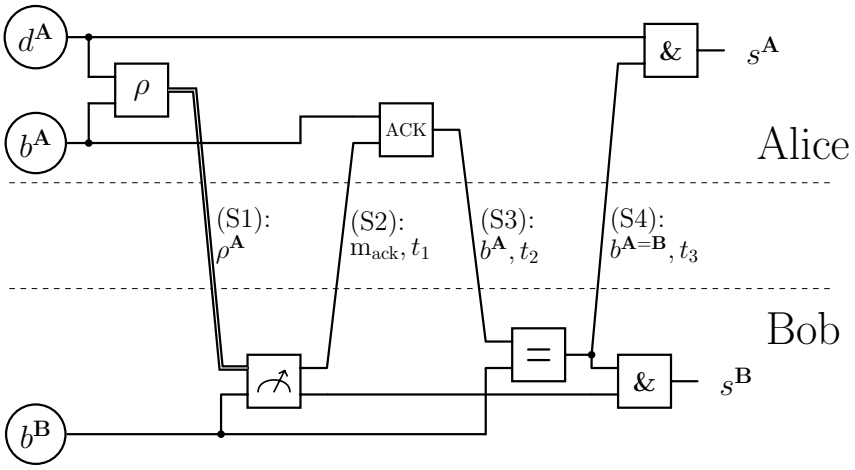

Figure 1. Protocol 1 (BB84, Quantum exchange and sifting only). Time flow is from left to right. Single (double) lines represent classical (quantum) communication. Local protocol actions are depicted by boxes: ρ depicts state preparation, the indicator is a quantum measurement device, the ACK box denotes that Alice waits for Bob's message until she continues with the protocol, = denotes the calculation of identical bases, & denotes the filtering of signals (in different bases).

the first (internal) stage of authentication throughout the complete chain of the QKD protocol. Therefore, in contrast to the case of authentication by universal hashing, now QKD post-processing protocols differing in the precise definition of their separate algorithmic steps (e.g. mode of authentication — immediate or delayed, exact order of exchange of sifting messages, whether error-correction bits are encrypted or not, etc.) become inequivalent and exhibit different types of vulnerabilities. For this reason each attack discussed below is adapted to a specific protocol. Both the protocols and the corresponding attacks are carefully and formally defined.

We consider exclusively but without loss of generality the case of BB84 QKD protocols, as the attacks we discuss are essentially independent of the particular form of quantum communication. Moreover, all protocols that we study are stated as prepare-and-measure ones. It is, however, straightforward to adapt the attacks discussed below to the case of entanglement based protocols.

It is implicitly assumed that on receiving messages Alice and Bob check their message tags for correctness, and that incorrect message tags lead them to conclude that Eve is intercepting, and to abort the protocol. In case the message authentication is UC-secure the resulting protocols are also UC-secure. A collection of used symbols is given in Table 1.

3.3. Protocol 1 – BB84 with immediate message authentication – Alice sends bases

We divide the protocol into two separate parts: (S) quantum state transmission and sifting, and (P) post processing (consisting of error correction, confirmation, and privacy amplification). Part (P) needs the result of (S) (i.e. the sifted keys) as input.

Table 1. Summary of symbols used in the paper.

Symbol	Description
$\mathbf{A}$, $\mathbf{B}$, $\mathbf{E}$	Legitimate parties: Alice, Bob; and eavesdropper Eve.
$\mathcal{Q}$, $\mathcal{C}$	quantum channel, classical channel
$b^{\mathbf{A}}$ ($b^{\mathbf{E}}$), $d^{\mathbf{A}}$ ($d^{\mathbf{E}}$)	Alice's (Eve's) string for bases choice and raw key, resp., used for preparing the quantum states.
$b^{\mathbf{B}}$, $d^{\mathbf{B}}$	Bob's bases choice and measurement results (i.e. his raw key).
$\rho^{\mathbf{A}}$ ($\rho^{\mathbf{E}}$)	quantum state, prepared by Alice (Eve).
$\mathrm{m}_{\mathrm{ack}}$	notification that a party has finished its measurements.
$g_K(\cdot)$	keyed hash function with key K.
$b^{\mathrm{X=Y}}$	string indicating the positions where the parties X and Y successfully prepared and measured in the same basis.
$s^{\mathbf{A}}$ ($s^{\mathbf{B}}$, $s^{\mathbf{E}}$)	sifted key of Alice (Bob, Eve).
$s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ ($s^{\mathbf{E}\leftrightarrow\mathbf{B}}$)	sifted key shared between Eve and Alice (Bob).
$\hat{s}^{\mathbf{B}}$	error corrected (reconciled) key of Bob.
$\hat{s}^{\mathbf{E}\leftrightarrow\mathbf{A}}$	error corrected (reconciled) key that Eve shares with Alice.
$K^{\mathbf{A}}$ ($K^{\mathbf{B}}$, $K^{\mathbf{E}}$)	final key of Alice (Bob, Eve).
$K^{\mathbf{E}\leftrightarrow\mathbf{A}}$ ($K^{\mathbf{E}\leftrightarrow\mathbf{B}}$)	final key shared between Eve and Alice (Bob).
$EC := \{EC_1, \ldots, EC_n\}$	set of predefined parity check matrices, used for forward error correction in different error rate regimes.
i	index into the set EC, denoting the actual parity check matrix EC_i used.
CO	description of (ITS) confirmation function.
P	description of (ITS) privacy amplification function.
ϵ	error rate on $\mathcal{Q}$.
fail	notification that a partner should abort protocol.

3.3.1. State transmission and sifting (S)
SUMMARY: 3 classical messages are exchanged. Each classical message is accompanied by a corresponding tag (keyed hash value, MAC).

1. *Setup.* $\mathbf{A}$ and $\mathbf{B}$ share the 3 keys K_1, K_2, K_3.
2. *Protocol messages.* Let $t_1 := g_{K_1}(\mathrm{m}_{\mathrm{ack}})$, $t_2 := g_{K_2}(b^{\mathbf{A}})$, and $t_3 := g_{K_3}(b^{\mathbf{A=B}})$ be the authentication tags used in messages (S2), (S3), and (S4), resp.
   (S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{B}$: $\rho^{\mathbf{A}}$
   (S2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $\mathrm{m}_{\mathrm{ack}}, t_1$
   (S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{A}}, t_2$
   (S4) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{A=B}}, t_3$
3. *Protocol actions.*

(Sa) **A** creates two random bit strings, her raw key $d^{\mathbf{A}}$, and the bases string $b^{\mathbf{A}}$, $d^{\mathbf{A}}, b^{\mathbf{A}} \in_r \{0,1\}^N$. For all pairs of bits $\left(d_k^{\mathbf{A}}, b_k^{\mathbf{A}}\right)$ **A** generates the corresponding quantum states $\rho_k^{\mathbf{A}} \in \{\rho^0, \rho^1, \rho^2, \rho^3\}$. Using $\mathcal{Q}$, **A** sends the quantum state $\rho^{\mathbf{A}} = \bigotimes_{k=1}^{N} \rho_k^{\mathbf{A}}$ ("string" of all $\rho_k^{\mathbf{A}}$'s), i.e. (S1) to **B**.

(Sb) **B** creates a random bases string $b^{\mathbf{B}} \in_r \{0,1\}^N$. **B** measures $\rho^{\mathbf{A}}$ in bases $b^{\mathbf{B}}$ and obtains $d^{\mathbf{B}} \in \{0, 1, \mathit{empty}\}^N$, where *empty* corresponds to no measurement result at **B**, e.g., due to absorption in the channel, or the imperfection of the detectors. For all k with $d_k^{\mathbf{B}} = \mathit{empty}$, **B** sets $b_k^{\mathbf{B}} = \mathit{empty}$.

(Sc) Using $\mathcal{C}$, **B** sends an acknowledgement message (S2) to **A**.

(Sd) **A** waits until she has received (S2), ensuring that the measurements have been finished before bases exchange is performed. Using $\mathcal{C}$, **A** sends (S3) to **B**.

(Se) **B** calculates a bit string $b^{\mathbf{A}=\mathbf{B}}$, such that $b_k^{\mathbf{A}=\mathbf{B}} = 1$, if $b_k^{\mathbf{A}} = b_k^{\mathbf{B}}$, and $b_k^{\mathbf{A}=\mathbf{B}} = 0$, otherwise, for $1 \leq k \leq N$. **B** removes from $d^{\mathbf{B}}$ all bits $d_k^{\mathbf{B}}$ where $b_k^{\mathbf{A}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{B}}$. Using $\mathcal{C}$, **B** sends (S4) to **A**.

(Sf) **A** removes from $d^{\mathbf{A}}$ all bits $d_k^{\mathbf{A}}$ where $b_k^{\mathbf{A}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{A}}$.

3.3.2. Post processing (P)

SUMMARY: 3 classical messages with MACs are exchanged.

1. *Setup.* **A** and **B** share 3 keys K_4, K_5, K_6.
2. *Protocol messages.* Let $T^{\mathbf{A}} = (i, EC_i(s^{\mathbf{A}}), CO, CO(s^{\mathbf{A}}))$.
   (P1) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}: \quad T^{\mathbf{A}}, g_{K_4}(T^{\mathbf{A}})$
   (P2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}: \quad \epsilon, g_{K_5}(\epsilon) \quad / \quad \mathit{fail}, g_{K_5}(\mathit{fail})$
   (P3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}: \quad P^{\mathbf{A}}, g_{K_6}(P^{\mathbf{A}}) \quad / \quad ---$
3. *Protocol actions.*

   (Pa) **A** estimates the parameters of $\mathcal{Q}$ (based on the error rate of previous rounds or by choosing a default value), selects a corresponding forward error correction algorithm EC_i from a predefined set, and calculates the syndrome $EC_i(s^{\mathbf{A}})$. **A** determines a confirmation function CO, and calculates $CO(s^{\mathbf{A}})$. **A** sends (P1).

   (Pb) **B** uses EC_i and $EC_i(s^{\mathbf{A}})$ to correct $s^{\mathbf{B}}$ resulting in $\hat{s}^{\mathbf{B}}$. **B** uses CO to calculate $CO(\hat{s}^{\mathbf{B}})$. **B** checks whether $CO(\hat{s}^{\mathbf{B}}) = CO(s^{\mathbf{A}})$. If the identity holds, **B** calculates the error rate ϵ and sends it to **A** (P2). If not, **B** sends *fail* to **A** (P2) and aborts the protocol.

   (Pc) If **A** receives ϵ, **A** determines a corresponding privacy amplification function $P^{\mathbf{A}}$, calculates $K^{\mathbf{A}} = P^{\mathbf{A}}(s^{\mathbf{A}})$, and sends (P3). If **A** receives *fail* she aborts the protocol.

   (Pd) If **B** has not aborted in step (Pb), he now calculates $K^{\mathbf{B}} = P^{\mathbf{A}}(\hat{s}^{\mathbf{B}})$. With probability almost 1 (determined by the confirmation function CO), $K^{\mathbf{A}} = K^{\mathbf{B}}$.

3.3.3. Attack against Protocol 1

Eve replaces the quantum channel between Alice and Bob with ideal quantum channels

and her instrumentation to prepare, store, and (almost) perfectly measure quantum states.
RESULT: Alice, Bob, and Eve share identical keys $K^{\mathbf{A}} = K^{\mathbf{B}} = K^{\mathbf{E}}$.

1. *Notation.*
   $\tilde{b}^x$: a string that deviates slightly from b^x to reach a hash collision with a given tag t [used in messages (S3') and (S4')].
2. *Protocol messages and messages inserted by Eve (marked by ').* Let $t_1 := g_{K_1}(\mathrm{m_{ack}})$, $t_2 := g_{K_2}(b^{\mathbf{A}})$, and $t_3 := g_{K_3}(b^{\mathbf{E}=\mathbf{B}})$ be the authentication tags used in messages (S2)–(S4).

   (S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{E}$: $\rho^{\mathbf{A}}$
   (S1') $\mathbf{E} \xrightarrow{\mathcal{Q}} \mathbf{B}$: $\rho^{\mathbf{E}}$
   (S2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $\mathrm{m_{ack}}, t_1$
   (S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $b^{\mathbf{A}}, t_2$
   (S3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $\tilde{b}^{\mathbf{E}}, t_2$
   (S4) $\mathbf{E} \xleftarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{E}=\mathbf{B}}, t_3$
   (S4') $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{E}$: $\tilde{b}^{\mathbf{A}=\mathbf{E}}, t_3$
   (P1) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}$: $T^{\mathbf{A}}, g_{K_4}(T^{\mathbf{A}})$
   (P2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $\epsilon, g_{K_5}(\epsilon)$ / $fail, g_{K_5}(fail)$
   (P3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}$: $P^{\mathbf{A}}, g_{K_6}(P^{\mathbf{A}})$ / —

3. *Protocol and attack actions.*

   (Sa) **A** performs step (Sa) of the protocol (prepares $\rho^{\mathbf{A}}$ and sends it in (S1)).
   (Sa') **E** intercepts (S1) from **A** and stores $\rho^{\mathbf{A}}$ in her quantum memory. Then **E** performs exactly as **A** in step (a) of the protocol: **E** determines random $d^{\mathbf{E}}$ and $b^{\mathbf{E}}$, prepares a state $\rho^{\mathbf{E}}$ and sends it in (S1') to **B**.
   (Sb) **B** performs step (Sb) of the protocol measuring the state **E** has prepared, $\rho^{\mathbf{E}}$, instead of $\rho^{\mathbf{A}}$, as in the protocol (in the following denoted as $\rho^{\mathbf{A}} \to \rho^{\mathbf{E}}$).
   (Sc) **B** performs step (Sc) of the protocol, i.e. he sends (S2).
   (Sd) **A** performs step (Sd) of the protocol. She sends (S3).
   (Sd') **E** intercepts (S3), i.e. $b^{\mathbf{A}}$ and the corresponding tag t_2, and measures her quantum memory in bases $b^{\mathbf{A}}$ and obtains an identical copy of **A**'s raw key, $d^{\mathbf{A}}$.
   (Sd") **E** determines $\tilde{b}^{\mathbf{E}}$ (e.g. using an exhaustive search), such that the intercepted t_2 validates $\tilde{b}^{\mathbf{E}}$ and $d_H(\tilde{b}^{\mathbf{E}}, b^{\mathbf{E}})$ is small (cf. Lemma 1), and sends (S3') to **B**.
   (Se) **B** performs step (Se) of the protocol ($b^{\mathbf{A}} \to \tilde{b}^{\mathbf{E}}$, $b^{\mathbf{A}=\mathbf{B}} \to b^{\mathbf{E}=\mathbf{B}}$), obtains $s^{\mathbf{B}}$ and sends message (S4).
   (Se') **E** intercepts (S4), i.e. $b^{\mathbf{E}=\mathbf{B}}$ and the corresponding tag t_3. **E** removes from $d^{\mathbf{E}}$ all bits $d_k^{\mathbf{E}}$ where $b_k^{\mathbf{E}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}$ (in general $s^{\mathbf{E}\leftrightarrow\mathbf{B}} \neq s^{\mathbf{B}}$ because **E** had to send $\tilde{b}^{\mathbf{E}}$ instead of her true basis choice $b^{\mathbf{E}}$ in step (Sd")).
   (Se") Using the algorithm detailed in Appendix B.1, **E** searches for a subsequence of $d^{\mathbf{A}}$ that coincides with $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ and calculates $b^{\mathbf{A}=\mathbf{E}}$ such that in **A**'s next step,

(Sf), **A** would create $s^{\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ as her sifted key. Typically **E** will have to allow for $O(\sqrt{n})$ bits that will be different between $s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ and $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ (see Lemma 2).

(Se"') As in step (Sd") **E** determines $\tilde{b}^{\mathbf{A}=\mathbf{E}}$ with small Hamming distance to $b^{\mathbf{A}=\mathbf{E}}$, this time validated by t_3 obtained in step (Se'), calculates the actual sifted key of **A**, $s^{\mathbf{E}\leftrightarrow\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ and sends (S4') to **A**.

(Sf) **A** performs step (Sf) of the protocol ($b^{\mathbf{A}=\mathbf{B}} \to \tilde{b}^{\mathbf{A}=\mathbf{E}}$) and obtains $s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}}$.

Note: Eve has almost reached her goal, as $s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}$ holds. The subsequent error correction step allows her to reach $K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}$:

(Pa) **A** performs step (Pa) of the protocol. Eve reads (P1), and uses the syndrome to correct her sifted key (in case **A**'s preparation and/or **E**'s quantum measurement and preparation are not 100% perfect, so that $s^{\mathbf{E}\leftrightarrow\mathbf{A}} \approx s^{\mathbf{A}}$).

(Pb) **B** performs step (Pb) of the protocol: $s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} = \hat{s}^{\mathbf{B}}$.

(Pc) **A** performs step (Pc) of the protocol and obtains $K^{\mathbf{A}} = P^{\mathbf{A}}(s^{\mathbf{A}})$.

(Pc') **E** reads (P3), the privacy amplification function $P^{\mathbf{A}}$. **E** calculates $K^{\mathbf{E}} = P^{\mathbf{A}}(s^{\mathbf{E}\leftrightarrow\mathbf{A}}) = K^{\mathbf{A}}$.

(Pd) **B** performs step (Pd) of the protocol: $K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}$.

This attack completely breaks protocol 1. Eve has an identical copy of Alice's and Bob's shared "secret" key. This is the strongest possible attack. For instance, using her copy of the key, Eve can simply decrypt messages from, and encrypt and/or authenticate new messages to both parties.

If this key is used to authenticate further QKD rounds, Eve can now continue with a much simpler impersonation attack, in which she does not have to calculate hash collisions or use her quantum memory.

3.4. Protocol 2 – BB84 with delayed message authentication – Alice sends bases

This protocol is very similar to Protocol 1, the difference is the authentication method: the authentication is delayed and performed only at the end of the protocol verifying the integrity of all messages. This, however, will change details of our attack strategy: until the very last message we don't have to care about authentication, but at the end we attack the privacy amplification matrix to get enough degrees of freedom to find collisions (step (Pc'), see below).

SUMMARY: 7 classical messages are exchanged. A nonce is used to enforce synchronization. The two last messages are authenticated with MACs.

1. *Notation.* $n^{\mathbf{B}}$: random number (nonce), created by **B**.
2. *Setup.* **A** and **B** share two keys K_1, K_2.
3. *Protocol messages.* Let $T^{\mathbf{A}} = (i, EC_i(s^{\mathbf{A}}), CO, CO(s^{\mathbf{A}}))$, $M^{\mathbf{A}} = (b^{\mathbf{A}}, T^{\mathbf{A}}, P^{\mathbf{A}})$, $M^{\mathbf{B}} = (n^{\mathbf{B}}, b^{\mathbf{A}=\mathbf{B}}, \epsilon/fail)$.

   (S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{B}$: $\rho^{\mathbf{A}}$

   (S2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $n^{\mathbf{B}}$

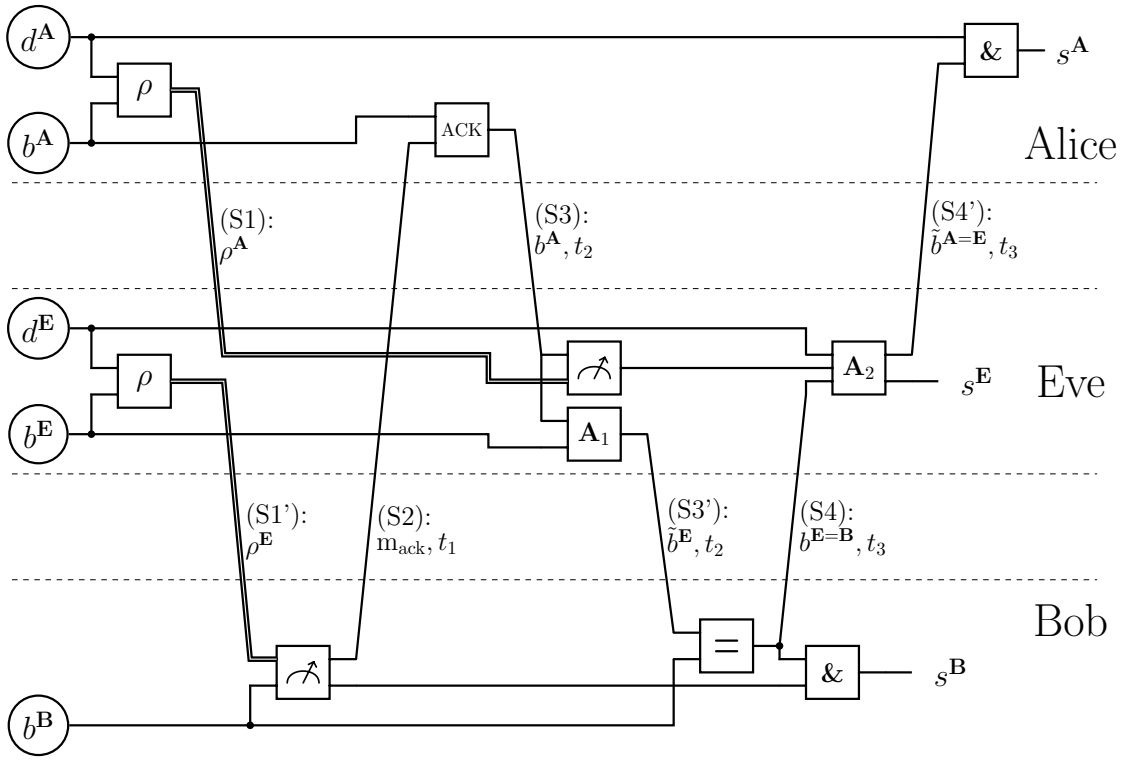

Figure 2. Interleaving attack against quantum exchange and sifting of Protocol 1, a QKD-protocol with immediate authentication. Time flow is from left to right. Single (double) lines represent classical (quantum) communication. See caption of Fig. 1 for a description of boxes and symbols. The new boxes $\mathrm{A}_{1,2}$ denote the attack actions, described in protocol steps (Se) through (Se'''). Employing quantum memory Eve manages to bring Alice and Bob to distill a sifted key that she knows with probability approaching 1.

(S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{A}}$
(S4) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{A=B}}$
(P1) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}$: $T^{\mathbf{A}}$
(P2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: ϵ / *fail*, $g_{K_1}(M^{\mathbf{B}})$
(P3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}$: $P^{\mathbf{A}}, g_{K_2}(M^{\mathbf{A}})$ / —

4. *Protocol actions.* Steps (Sa)–(Sf) and (Pa)–(Pd) are identical to that of protocol 1, with the following exceptions: (a) only the two last messages of the protocol, (P2) and (P3), [which are sent in step (Pb) and (Pc)] have MACs attached that authenticate all messages from Bob to Alice and Alice to Bob, respectively, (b) in step (Sc) the message (S2) contains a nonce $n^{\mathbf{B}}$, a random number that is chosen by Bob and used to ensure that Bob has finished measuring before the bases exchange starts. Using a fixed $\mathrm{m}_{\mathrm{ack}}$ as in protocol 1 instead of the random nonce $n^{\mathbf{B}}$ would allow for a trivial attack.

3.4.1. Attack against Protocol 2 (Eve only attacks messages to Bob) Eve replaces the quantum channel between Alice and Bob, with ideal quantum channels and her instrumentation to prepare, store and perfectly measure quantum states. The first part of the attack is similar to the attack against protocol 1 but it differs in several essential instances. All steps from (Sa) to (Sd') are basically the same, but messages (S2) and (S3) are sent without MACs. From now on the attack differs so that Eve can cope with the form of postponed authentication utilized in protocol 2. In particular, we assume that Eve cannot manipulate the message that contains the error rate ϵ on the quantum channel. This could be the case, for example, if ϵ is encoded as 16 bit integer: the existence of hash collisions is very unlikely, since it is impossible to reach the needed Hamming distance of at least 19 (see Sec. 3.1). This in turn implies, that Eve can also not manipulate any previous message from Bob to Alice (since she does not know what value of ϵ Bob will be transmitting, she does not know which messages to prepare to get a hash collision). In particular, Eve cannot modify the sifting message of Bob, which rules out an attack analogous to the attack against protocol 1, described above. Amazingly, although Eve cannot modify any message from Bob to Alice, she can still mount the most powerful attack (Alice, Bob, and Eve share the same key)!
RESULT: Alice, Bob, and Eve share identical keys $K^{\mathbf{A}} = K^{\mathbf{B}} = K^{\mathbf{E}}$.

1. *Protocol messages and messages inserted by Eve (marked by ').* In addition to the definitions in the protocol above, let $t_2 = g_{K_2}(M^{\mathbf{A}})$.

(S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{E}$: $\rho^{\mathbf{A}}$
(S1') $\mathbf{E} \xrightarrow{\mathcal{Q}} \mathbf{B}$: $\rho^{\mathbf{E}}$
(S2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $n^{\mathbf{B}}$
(S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $b^{\mathbf{A}}$
(S3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{E}}$
(S4) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{E}=\mathbf{B}}$
(P1) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $T^{\mathbf{A}}$
(P1') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $T^{\mathbf{E}}$
(P2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: ϵ / *fail*, $g_{K_1}(M^{\mathbf{B}})$
(P3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $P^{\mathbf{A}}, t_2$ / —
(P3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $P^{\mathbf{E}}, t_2$ / —

2. *Protocol and attack actions.*

(Sa) – (Sd') Identical to those of protocol 1 (cf. Sec. 3.3.3), up to the absence of authentication tags in the present protocol.
(Sd") **E** performs step (Sd) of the protocol ($b^{\mathbf{A}} \to b^{\mathbf{E}}$) and sends message (S3') to **B**.
(Se) **B** performs step (Se) of the protocol ($b^{\mathbf{A}} \to b^{\mathbf{E}}$), obtains $b^{\mathbf{E}=\mathbf{B}}$ and $s^{\mathbf{B}}$, and sends message (S4).
(Se') **E** reads message (S4), i.e. $b^{\mathbf{E}=\mathbf{B}}$. She removes from $d^{\mathbf{E}}$ all bits $d_k^{\mathbf{E}}$ for $k : b_k^{\mathbf{E}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}$, possibly with noise.
(Sf) **A** performs step (Sf) of the protocol ($b^{\mathbf{A}=\mathbf{B}} \to b^{\mathbf{E}=\mathbf{B}}$) and obtains $s^{\mathbf{A}}$.

(Sf') **E** removes from the string $d^{\mathbf{A}}$ (which she knows exactly) all bits $d_k^{\mathbf{E}}$ for $k : b_k^{\mathbf{E}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{E}\leftrightarrow\mathbf{A}} = s^{\mathbf{A}}$.

Note: Eve now shares two keys with Alice and Bob respectively $s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ and $s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}$ (or $s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}$ as discussed above) but these keys are not correlated. After the subsequent error correction step **E** already shares $\hat{s}^{\mathbf{A}} = \hat{s}^{\mathbf{E}\leftrightarrow\mathbf{A}}$ and $\hat{s}^{\mathbf{E}\leftrightarrow\mathbf{B}} = \hat{s}^{\mathbf{B}}$. Finally, attacking the privacy amplification step of the protocol **E** succeeds in achieving her ultimate goal $K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}$:

(Pa) **A** performs this step in the protocol and sends message (P1).

(Pa') **E** intercepts (P1), produces $T^{\mathbf{E}} = (i, EC_i(s^{\mathbf{E}\leftrightarrow\mathbf{B}}), CO, CO(s^{\mathbf{E}\leftrightarrow\mathbf{B}}))$ and sends message (P1') to **B**. (If **E** would anticipate an error between her and **B** that is too low, she can artificially modify her sifted key $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ to increase the error that **B** registers.)

(Pb) **B** performs step (Pb) of the protocol ($T^{\mathbf{A}} \rightarrow T^{\mathbf{E}}$), obtains $\hat{s}^{\mathbf{B}} = s^{\mathbf{E}\leftrightarrow\mathbf{B}}$, calculates the error rate, determines $M^{\mathbf{B}} = (n^{\mathbf{B}}, b^{\mathbf{E}=\mathbf{B}}, \epsilon/\mathit{fail})$, where $b^{\mathbf{A}=\mathbf{B}} \rightarrow b^{\mathbf{E}=\mathbf{B}}$ and sends message (P2).

(Pc) **A** accepts the authenticity of all the messages she has received, i.e. (S2), (S4), (P2), since **E** has not modified any message and performs step (Pc) sending (P3).

(Pc') **E** intercepts (P3). To break the authentication of (P3), **E** calculates another PA function $P^{\mathbf{E}}$, such that $P^{\mathbf{E}}(s^{\mathbf{E}\leftrightarrow\mathbf{B}}) = K^{\mathbf{A}}$ and $t_2 = g_{K_2}(b^{\mathbf{E}}, T^{\mathbf{E}}, P^{\mathbf{E}})$. To ensure the last condition it is sufficient that the message $(b^{\mathbf{E}}, T^{\mathbf{E}}, P^{\mathbf{E}}) = M^{\mathbf{E}}$ collides with $M^{\mathbf{A}}$ under the inner authentication hash function f, i.e $f(M^{\mathbf{E}}) = f(M^{\mathbf{A}})$. **E** sends (P3') to **B**. (If Eve would be satisfied with Alice and Bob having different keys, both of which she knows, Eve only searches for any PA function $P^{\mathbf{E}}$ such that $f(M^{\mathbf{E}}) = f(M^{\mathbf{A}})$, but accepts $K^{\mathbf{B}} = P^{\mathbf{E}}(s^{\mathbf{E}\leftrightarrow\mathbf{B}}) = K^{\mathbf{E}\leftrightarrow\mathbf{B}} \neq K^{\mathbf{A}} = P^{\mathbf{A}}(s^{\mathbf{E}\leftrightarrow\mathbf{A}})$.)

(Pd) **B** accepts the authenticity of all the messages he has received, i.e. (S3'), (P1'), (P3'), since he has received a valid tag (t_2) and performs the final step of the protocol to get $K^{\mathbf{B}} = P^{\mathbf{E}}(\hat{s}^{\mathbf{B}}) = K^{\mathbf{E}} = K^{\mathbf{A}}$.

3.5. Protocol 3 – BB84 with immediate message authentication – Bob sends bases

This protocol is a variant of protocol 1, also using immediate message authentication. Only part (S), i.e. the quantum state transmission and sifting is different: After measuring the quantum signals, instead of sending an acknowledge message as in protocol 1, Bob sends his bases information to Alice (implicitly acknowledging that he has finished his measurements). Alice replies with her basis information.

3.5.1. State transmission and sifting

SUMMARY: 2 classical messages with MACs are exchanged.

1. *Setup.* **A** and **B** share two keys K_1, K_2.

2. *Protocol messages.*
   (S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{B}:$ $\rho^{\mathbf{A}}$
   (S2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}:$ $b^{\mathbf{B}}, g_{K_1}(b^{\mathbf{B}})$
   (S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{B}:$ $b^{\mathbf{A}=\mathbf{B}}, g_{K_2}(b^{\mathbf{A}=\mathbf{B}})$
3. *Protocol actions.*
   (Sa) same as (Sa) in protocol 1: **A** creates two random bit strings, $d^{\mathbf{A}}, b^{\mathbf{A}} \in_r \{0,1\}^N$. For each pair $\left(d_k^{\mathbf{A}}, b_k^{\mathbf{A}}\right)$ **A** generates the corresponding quantum state $\rho_k^{\mathbf{A}} \in \{\rho^0, \rho^1, \rho^2, \rho^3\}$. Using $\mathcal{Q}$, **A** sends the quantum state $\rho^{\mathbf{A}} = \bigotimes_{k=1}^{N} \rho_k^{\mathbf{A}}$ ("string" of all $\rho_k^{\mathbf{A}}$'s), i.e. (S1), to **B**.
   (Sb) same as (Sb) in protocol 1: **B** creates a random bit string $b^{\mathbf{B}} \in_r \{0,1\}^N$. **B** measures $\rho^{\mathbf{A}}$ in bases $b^{\mathbf{B}}$ and obtains $d^{\mathbf{B}} \in \{0, 1, \mathit{empty}\}^N$ as result. For all k with $d_k^{\mathbf{B}} = \mathit{empty}$, **B** sets $b_k^{\mathbf{B}} = \mathit{empty}$.
   (Sc) Using $\mathcal{C}$, **B** sends (S2), i.e. $b^{\mathbf{B}}$, to **A**.
   (Sd) **A** waits until she has received (S2). **A** calculates the bit string $b^{\mathbf{A}=\mathbf{B}}$, such that $b_k^{\mathbf{A}=\mathbf{B}} = 1$ if $b_k^{\mathbf{A}} = b_k^{\mathbf{B}}$, and $b_k^{\mathbf{A}=\mathbf{B}} = 0$, otherwise. **A** removes from $d^{\mathbf{A}}$ all bits $d_k^{\mathbf{A}}$ where $b_k^{\mathbf{A}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{A}}$.
   (Se) Using $\mathcal{C}$, **A** sends (S3), i.e. $b^{\mathbf{A}=\mathbf{B}}$, to **B**.
   (Sf) **B** removes from $d^{\mathbf{B}}$ all bits $d_k^{\mathbf{B}}$ where $b_k^{\mathbf{A}=\mathbf{B}} = 0$ and obtains $s^{\mathbf{B}}$.

3.5.2. Post processing (P)

This part is completely identical to part (P) of protocol 1, cf. Sec. 3.3.2.

3.5.3. Attack against Protocol 3

Eve replaces the quantum channel between Alice and Bob, with ideal quantum channels and her instrumentation. Eve must be able to prepare and perfectly measure quantum states. She does not need a quantum memory to perform her attack. Essentially this attack is a modified version of the well known intercept-resend attack, whereby the currently discussed authentication mechanism allows Eve to conceal the difference between the sifted keys of Alice and Bob (of roughly 25%) in the postprocessing stage of the protocol.
RESULT: Alice, Bob, and Eve share identical keys $K^{\mathbf{A}} = K^{\mathbf{B}} = K^{\mathbf{E}}$.

1. *Notation.*
   $\tilde{b}^x$: a string that deviates slightly from b^x to reach a hash collision with a given tag t [used in messages (S2') and (S3')].
2. *Protocol messages and messages inserted by Eve (marked by ').* Let $t_1 = g_{K_1}(b^{\mathbf{B}})$, $t_2 = g_{K_2}(b^{\mathbf{A}=\mathbf{E}})$, $t_3 = g_{K_3}(T^{\mathbf{A}})$, $t_5 = g_{K_5}(P^{\mathbf{A}})$.
   (S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{E}:$ $\rho^{\mathbf{A}}$
   (S1') $\mathbf{E} \xrightarrow{\mathcal{Q}} \mathbf{B}:$ $\rho^{\mathbf{E}}$
   (S2) $\mathbf{E} \xleftarrow{\mathcal{C}} \mathbf{B}:$ $b^{\mathbf{B}}, t_1$

(S2') $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{E}$: $\tilde{b}^{\mathbf{E}}, t_1$
(S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $b^{\mathbf{A}=\mathbf{E}}, t_2$
(S3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $\tilde{b}^{\mathbf{E}=\mathbf{B}}, t_2$
(P1) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $T^{\mathbf{A}}, t_3$
(P1') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $T^{\mathbf{E}}, t_3$
(P2) $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{B}$: $\epsilon, g_{K_4}(\epsilon)$ / $fail, g_{K_4}(fail)$
(P3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $P^{\mathbf{A}}, t_5$ / —
(P3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $P^{\mathbf{E}}, t_5$ / —

3. *Protocol and attack actions.*

(Sa) **A** performs step (Sa) of the protocol.

(Sa') **E** creates a random bit strings, $b^{\mathbf{E}} \in_r \{0,1\}^N$. **E** intercepts (S1) from **A** and measures $\rho^{\mathbf{A}}$ in bases $b^{\mathbf{E}}$, she obtains $d^{\mathbf{E}}$. For each pair $\left(d_k^{\mathbf{E}}, b_k^{\mathbf{E}}\right)$, **E** prepares the corresponding quantum state $\rho_k^{\mathbf{E}}$ and sends (S1') to **B**.

(Sb) **B** performs step (Sb) of the protocol ($\rho^{\mathbf{A}} \to \rho^{\mathbf{E}}$).

(Sc) **B** performs step (Sc) of the protocol, i.e. he sends (S2).

(Sd') **E** intercepts (S2) and performs **A**'s step (Sd) of the protocol($b^{\mathbf{A}=\mathbf{B}} \to b^{\mathbf{E}=\mathbf{B}}$, $b^{\mathbf{A}} \to b^{\mathbf{E}}$) and obtains her sifted key with Bob, $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$.

(Sc') **E** calculates $\tilde{b}^{\mathbf{E}}$, such that the intercepted t_1 validates $\tilde{b}^{\mathbf{E}}$ and $d_H(\tilde{b}^{\mathbf{E}}, b^{\mathbf{E}})$ is small. She then performs **B**'s step (Sc) of the protocol ($b^{\mathbf{B}} \to \tilde{b}^{\mathbf{E}}$), i.e. she sends (S2') to **A**.

(Sd) **A** performs step (Sd) of the protocol ($b^{\mathbf{B}} \to \tilde{b}^{\mathbf{E}}$, $b^{\mathbf{A}=\mathbf{B}} \to b^{\mathbf{A}=\mathbf{E}}$), she obtains $b^{\mathbf{A}=\mathbf{E}}$ (which is defined by $b_k^{\mathbf{A}=\mathbf{E}} = 1$, if $b_k^{\mathbf{A}} = \tilde{b}_k^{\mathbf{E}}$, and $b_k^{\mathbf{A}=\mathbf{E}} = 0$, otherwise) and $s^{\mathbf{A}}$.

(Se) **A** performs step (Se) of the protocol ($b^{\mathbf{A}=\mathbf{B}} \to b^{\mathbf{A}=\mathbf{E}}$), i.e. she sends (S3).

(Sf') **E** intercepts (S3) and performs **B**'s step (Sf) of the protocol ($d^{\mathbf{B}} \to d^{\mathbf{E}}$, $b^{\mathbf{A}=\mathbf{B}} \to b^{\mathbf{A}=\mathbf{E}}$) and obtains (approximately) her sifted key with **A**, $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$. (There are small deviations between $s^{\mathbf{A}}$ and $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ since **E** had to send $\tilde{b}^{\mathbf{E}}$ instead of $b^{\mathbf{E}}$).

(Se') **E** determines the string $b^{\mathbf{E}=\mathbf{B}}$, such that $b_k^{\mathbf{E}=\mathbf{B}} = 1$, if $b_k^{\mathbf{E}} = b_k^{\mathbf{B}}$, and $b_k^{\mathbf{E}=\mathbf{B}} = 0$, otherwise. **E** then calculates the string $\tilde{b}^{\mathbf{E}=\mathbf{B}}$, such that the intercepted t_2 validates $\tilde{b}^{\mathbf{E}=\mathbf{B}}$ and $d_H(\tilde{b}^{\mathbf{E}=\mathbf{B}}, b^{\mathbf{E}=\mathbf{B}})$ is small. Now **E** performs **A**'s step (Se) of the protocol ($b^{\mathbf{A}=\mathbf{B}} \to \tilde{b}^{\mathbf{E}=\mathbf{B}}$), i.e. she sends (S3').

(Sf) **B** performs step (Sf) of the protocol ($b^{\mathbf{A}=\mathbf{B}} \to \tilde{b}^{\mathbf{E}=\mathbf{B}}$), and obtains his sifted key, $s^{\mathbf{B}}$ (there are small deviations between $s^{\mathbf{B}}$ and $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ since **E** had to send $\tilde{b}^{\mathbf{E}=\mathbf{B}}$ instead of $b^{\mathbf{E}=\mathbf{B}}$).

Note: Now Eve possesses almost identical copies of Alice's and Bob's keys, respectively: $s^{\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ and $s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}$ (while $s^{\mathbf{A}}$ and $s^{\mathbf{B}}$ will differ in approximately 25% of the bits due to Eve's quantum intercept-resend attack). The subsequent steps allow Eve to transform her key $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ into $s^{\mathbf{A}}$ and make Bob transform his key $s^{\mathbf{B}}$ into a new key $\hat{s}^{\mathbf{B}}$, which she knows:

(Pa) **A** performs step (Pa) of the protocol, i.e. she sends (P1).

(Pb') **E** performs **B**'s step (Pb) of the protocol, i.e. she intercepts (P1) to learn the syndrome $EC_i(s^{\mathbf{A}})$, and corrects her sifted key $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$ to $s^{\mathbf{A}}$.

(Pa') **E** performs **A**'s step (Pa) of the protocol, but modifies her key $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ such that $EC^{\mathbf{E}}(s^{\mathbf{E}\leftrightarrow\mathbf{B}})$ will allow **B** to correct his sifted key to the modified $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ and that the resulting (P1'), i.e. $T^{\mathbf{E}} = (i, EC_i(s^{\mathbf{E}\leftrightarrow\mathbf{B}}), CO, CO(s^{\mathbf{E}\leftrightarrow\mathbf{B}}))$, is compatible with tag t_3. **E** sends (P1').

(Pb) **B** performs step (Pb) of the protocol, i.e. he corrects his sifted key $s^{\mathbf{B}}$ and obtains $\hat{s}^{\mathbf{B}}$. Now Eve shares $s^{\mathbf{A}}$ with Alice, and $\hat{s}^{\mathbf{B}}$ with Bob.

(Pc) **A** performs step (Pc) of the protocol, i.e. she determines a privacy amplification function $P^{\mathbf{A}}$, applies it to her sifted key, and obtains $K^{\mathbf{A}} = P^{\mathbf{A}}(s^{\mathbf{A}})$. **A** sends (P3).

(Pc') **E** intercepts (P3) to learn the privacy amplification function $P^{\mathbf{A}}$ and thus **A**'s final key $K^{\mathbf{A}}$. **E** calculates another PA function $P^{\mathbf{E}}$ such that $P^{\mathbf{E}}(\hat{s}^{\mathbf{B}}) = K^{\mathbf{A}}$ and that (P3') is compatible with tag t_5.

(Pd) **B** performs step (Pd) of the protocol, i.e. he applies $P^{\mathbf{E}}$ and gets $K^{\mathbf{B}} = P^{\mathbf{E}}(\hat{s}^{\mathbf{B}}) = K^{\mathbf{A}}$.

Again, Eve managed to break the protocol completely, as she knows Alice's and Bob's shared "secret" key.

3.6. Implications of protocol modifications on the presented attacks

3.6.1. No separate step for transmitting the privacy amplification function In [20, p. 83] it has been proposed that the privacy amplification function $P^{\mathbf{A}}$ is not transmitted in a separate protocol step (our step (P3)), but can be constructed from previously exchanged basis information ([3] uses this method to counter the attack described in [4]). However, no strict security proof of the resulting protocol has ever been put forward.

For the discussed two-step authentication our attack against protocol 1 still works without step (P3) since we don't attack the post processing step at all. Also the attack against protocol 3 still works without step (P3), but is not so powerful. Since Eve has complete knowledge of the basis information, she can just apply the respective PA function individually to her keys with Alice and Bob. Consequently, Eve will know Alice's and Bob's final keys which will be, however, different.

The case of protocol 2 is slightly more complicated but the outcome is identical to that of protocol 3. In this case the last communication message from Alice to Bob is (P1), and, naturally, it has to be extended to carry also the authentication tag $t_2 = g_{K_2}(M^{\mathbf{A}})$, whereby now $M^{\mathbf{A}} = (b^{\mathbf{A}}, T^{\mathbf{A}})$. Eve will have to modify her attack. Now she has to look for an error correction syndrome $T^{\mathbf{E}}$, so that $M^{\mathbf{E}} = (b^{\mathbf{E}}, T^{\mathbf{E}})$, collides with $M^{\mathbf{A}}$ under the inner authentication hash function f, i.e $f(M^{\mathbf{E}}) = f(M^{\mathbf{A}})$. To do so Eve is free to modify her sifted key $s^{\mathbf{E}\leftrightarrow\mathbf{B}} \rightarrow \tilde{s}^{\mathbf{E}\leftrightarrow\mathbf{B}}$, so that $T^{\mathbf{E}} = (i, EC_i(\tilde{s}^{\mathbf{E}\leftrightarrow\mathbf{B}}), CO, CO(\tilde{s}^{\mathbf{E}\leftrightarrow\mathbf{B}}))$ would ensure the required collision. As in the case of protocol 3 Eve has complete knowledge of the bases of Alice and Bob. She can again apply the respective PA

functions independently and obtain the final keys of Alice and Bob, which differ one from the other.

3.6.2. One-time pad encryption of the error correction syndrome Ref. [21] presented a protocol in which parity bits are encrypted with a one-time pad (using key that is preshared or generated in previous rounds). Since Alice and Bob use in addition a (large) key which is not known to Eve, one could expect that attacks will be impossible. Nevertheless, we will briefly outline modified attacks against such a protocol.

If Eve uses a quantum memory in her attack she will learn Alice's complete sifted key. Therefore, she can calculate the exact syndrome, that Alice will OTP-encrypt and send. From the plain and encrypted syndrome, Eve gets the one-time pad, encrypts *her* syndrome with it and continues the attack.

If Eve performs an attack without quantum memory, her and Alice's sifted key will differ in a small number of bits (the Hamming distance w of the two keys), the positions of which are known to Eve. Thus Eve can create the set of all possible sifted keys of Alice of size 2^w, which is only a very small subset of all possible keys of length approximately $n/2$, and is also smaller than the set of all possible message tags. Then Eve decides randomly to take one element of this set to be Alice's sifted key. Compared to a guess without previous knowledge she could dramatically increase her chances of guessing correctly, although the probability is still quite low, i.e. $p = 2^{-w}$. Assuming she has guessed correctly, she can now calculate the syndrome that Alice has sent, and thus get also the one-time pad. She uses it then for encrypting the syndrome that she sends to Bob.

3.7. Overview of attack approaches for adversaries with and without quantum memory

Up to now we have presented three attacks in which Eve on receiving a protocol message from Alice (Bob) sends either the original message or a modified one to Bob (Alice). In Sec. 3.8 we will present a different kind of attack. The attacks presented so far are not isolated cases of adversary success strategies in the case of weak authentication that uses the approach of Ref. [3]. The attacks are actually made up of building blocks that can be combined and applied in a wide variety of settings. We illustrate this fact by presenting a systematic overview of successful attacks against a range of protocols comprising the cases of sifting being started by Alice or Bob, authentication being immediate or delayed. Moreover for all the cases we distinguish between two levels of adversary resources: i) "classical only", i.e. sufficiently high computing power or ii) "quantum and classical", i.e. a combination of quantum resources (quantum memory) and classical ones (as in i)). These attacks are summarized in Tables 2 and 3. The attacks are not described in full detail and the tables focus on the adversary activities alone. The full attacks, can however be easily deduced by comparing the table contents referring to Attacks 1, 2 and 3 with the detailed description for these cases, given above.

Furthermore, using arguments similar to those presented in Section 3.6 one can

Table 2. Overview of attacks against the sifting stage of different protocol variants. QM denotes whether Eve uses a quantum memory in her attack. The notation "Protocol 1–3" refers to the protocols and corresponding attacks described in full detail above. $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ denotes the quantum state which encodes the bit string $d^{\mathbf{E}}$ in bases $b^{\mathbf{E}}$. $\approx$ denotes that two sifted keys deviate only weakly (error correction can reconcile them). $\not\approx$ denotes a deviation of two sifted keys by typically 25%. If not otherwise stated, **E** performs sifting with the appropriate bases of **A** and **B**.

	QM	Immediate Authentication	Delayed Authentication†
A sends bases	Y	*Protocol 1–Interleaving attack:* **E** stores $\rho^{\mathbf{A}}$ in quantum memory, **E** sends random $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** substitutes $\tilde{b}^{\mathbf{E}}$ for $b^{\mathbf{A}}$, **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{A}}$ and learns $d^{\mathbf{A}}$, **E** calculates $\tilde{b}^{\mathbf{A}=\mathbf{E}}$ to force $s^{\mathbf{E}\leftrightarrow\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{B}}$, **E** substitutes $\tilde{b}^{\mathbf{A}=\mathbf{E}}$ for $b^{\mathbf{E}=\mathbf{B}}$. $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}}$ (Case 1)	*Protocol 2–Interleaving attack:* **E** stores $\rho^{\mathbf{A}}$ in quantum memory, **E** sends random $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** substitutes $b^{\mathbf{E}}$ for $b^{\mathbf{A}}$, **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{A}}$ and learns $d^{\mathbf{A}}$, **E** listens to $b^{\mathbf{E}=\mathbf{B}}$ (no substitution!). $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ (Case 2)
	N	*Intercept-resend attack:* **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{E}}$ and gets $d^{\mathbf{E}}$, **E** sends $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** substitutes $\tilde{b}^{\mathbf{E}}$ for $b^{\mathbf{A}}$, **E** substitutes $\tilde{b}^{\mathbf{A}=\mathbf{E}}$ for $b^{\mathbf{A}=\mathbf{B}}$. $\boxed{s^{\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}}$ (Case 3)	*One-sided intercept-resend attack:* **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{E}}$ and gets $d^{\mathbf{E}}$, **E** sends $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** substitutes $b^{\mathbf{E}}$ for $b^{\mathbf{A}}$, **E** listens to $b^{\mathbf{E}=\mathbf{B}}$ (no substitution!). $\boxed{s^{\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ (Case 4)
B sends bases	Y	*Interleaving attack:* **E** stores $\rho^{\mathbf{A}}$ in quantum memory, **E** sends random $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** listens to $b^{\mathbf{B}}$ (no substitution!), **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{B}}$, **E** listens to $b^{\mathbf{A}=\mathbf{B}}$, determines $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$, **E** substitutes $\tilde{b}^{\mathbf{E}=\mathbf{B}}$ for $b^{\mathbf{A}=\mathbf{B}}$. $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}}$ (Case 3)	*Interleaving attack:* **E** stores $\rho^{\mathbf{A}}$ in quantum memory, **E** sends random $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** listens to $b^{\mathbf{B}}$ (no substitution!), **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{B}}$, **E** listens to $b^{\mathbf{A}=\mathbf{B}}$, determines $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$, **E** substitutes $b^{\mathbf{E}=\mathbf{B}}$ for $b^{\mathbf{A}=\mathbf{B}}$. $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ (Case 2)
	N	*Protocol 3–Intercept-resend attack:* **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{E}}$ and gets $d^{\mathbf{E}}$, **E** sends $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** substitutes $\tilde{b}^{\mathbf{E}}$ for $b^{\mathbf{B}}$, **E** substitutes $\tilde{b}^{\mathbf{E}=\mathbf{B}}$ for $b^{\mathbf{A}=\mathbf{B}}$. $\boxed{s^{\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}}$ (Case 3)	*One-sided intercept-resend attack:* **E** measures $\rho^{\mathbf{A}}$ in $b^{\mathbf{E}}$ and gets $d^{\mathbf{E}}$, **E** sends $\rho^{\mathbf{E}}(b^{\mathbf{E}}, d^{\mathbf{E}})$ to **B**, **E** listens to $b^{\mathbf{B}}$ (no substitution!), **E** substitutes $b^{\mathbf{E}=\mathbf{B}}$ for $b^{\mathbf{A}=\mathbf{B}}$. $\boxed{s^{\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ (Case 4)

†In these cases **E** does not substitute messages from **B** to **A**.

Table 3. Overview of four attack classes against the protocol stages after sifting. The attacks pertain to the output of sifting, which according to Table 2, yields four different types of correlations between the sifted keys of **A**, **E**, and **B**: two for immediate (cases 1, 3) and two for delayed authentication (cases 2, 4), respectively. Note, that for the sake of simplicity we do not use the "hat" notation for error corrected keys.

Immediate Authentication	Delayed Authentication[†]
$\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}}$ (Case 1) EC: **E** does nothing. result: $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} = s^{\mathbf{B}}}$ PA: **E** listens to the PA function $P^{\mathbf{A}}$, **E** calculates $K^{\mathbf{E}} := P^{\mathbf{A}}(s^{\mathbf{E}\leftrightarrow\mathbf{A}})$. result: $\boxed{K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}}$	$\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ (Case 2) EC: **E** intercepts $T^{\mathbf{A}}$, **E** calculates $T^{\mathbf{E}}$, **E** sends $T^{\mathbf{E}}$ to **B**. result: $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ PA: **E** intercepts the PA function $P^{\mathbf{A}}$, **E** calculates $K^{\mathbf{E}} := P^{\mathbf{A}}(s^{\mathbf{E}\leftrightarrow\mathbf{A}})$, **E** calculates new PA function $P^{\mathbf{E}}$, **E** sends $P^{\mathbf{E}}$ to **B**. result: $\boxed{K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}}$
$\boxed{s^{\mathbf{A}} \approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} \approx s^{\mathbf{B}}}$ (Case 3) EC: **E** intercepts $T^{\mathbf{A}}$, **E** corrects $s^{\mathbf{E}\leftrightarrow\mathbf{A}}$, obtains $s^{\mathbf{A}}$, **E** modifies $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$, calculates $T^{\mathbf{E}}$, **E** sends $T^{\mathbf{E}}$ to **B**. result: $\boxed{s^{\mathbf{A}} = s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ PA: **E** intercepts the PA function $P^{\mathbf{A}}$, **E** calculates $K^{\mathbf{E}} := P^{\mathbf{A}}(s^{\mathbf{E}\leftrightarrow\mathbf{A}})$, **E** calculates new PA function $P^{\mathbf{E}}$, **E** sends $P^{\mathbf{E}}$ to **B**. result: $\boxed{K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}}$	$\boxed{s^{\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ (Case 4) EC: **E** intercepts $T^{\mathbf{A}}$, **E** calculates $T^{\mathbf{E}}$, **E** sends $T^{\mathbf{E}}$ to **B**. result: $\boxed{s^{\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{A}} \not\approx s^{\mathbf{E}\leftrightarrow\mathbf{B}} = s^{\mathbf{B}}}$ PA: **E** intercepts the PA function $P^{\mathbf{A}}$, **E** calculates $K^{\mathbf{E}} := P^{\mathbf{A}}(s^{\mathbf{E}\leftrightarrow\mathbf{A}})$, **E** calculates new PA function $P^{\mathbf{E}}$, **E** sends $P^{\mathbf{E}}$ to **B**. result: $\boxed{K^{\mathbf{A}} \neq K^{\mathbf{E}} = K^{\mathbf{B}}}$

[†]In these cases **E** does not substitute messages from **B** to **A**.

construct attacks against modified versions of these protocols, including encryption of error-correction information and reuse of common, sifting-stage randomness for privacy amplification without communication.

3.8. Another attack against Protocol 2 (Eve attacks in both directions)

In our previous attacks Eve substitutes certain messages but sticks to the original message order of the protocol. In the following attack Eve exchanges a sequence of messages with Alice first. When she needs to send an authentication tag to Alice, she starts her communication with Bob and continues until she obtains the necessary tag

from him. Then Eve continues her communication with Alice.

In contrast to the previous attack against protocol 2 (cf. Sec. 3.4.1) this attack allows Eve to modify also messages that are sent to Alice.

1. *Protocol messages and messages inserted by Eve (marked by ').* Let $t_1 := g_{K_1}(M^{\mathbf{B}})$, and remember that $t_2 := g_{K_2}(M^{\mathbf{A}})$.

(S1) $\mathbf{A} \xrightarrow{\mathcal{Q}} \mathbf{E}$: $\rho^{\mathbf{A}}$
(S2') $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{E}$: $n^{\mathbf{E}}$
(S3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $b^{\mathbf{A}}$
(S1') $\mathbf{E} \xrightarrow{\mathcal{Q}} \mathbf{B}$: $\rho^{\mathbf{E}} = \rho^{\mathbf{A}}$
(S2) $\mathbf{E} \xleftarrow{\mathcal{C}} \mathbf{B}$: $n^{\mathbf{B}}$
(S3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{E}} = b^{\mathbf{A}}$
(S4) $\mathbf{E} \xleftarrow{\mathcal{C}} \mathbf{B}$: $b^{\mathbf{E}=\mathbf{B}}$
(P1') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $T^{\mathbf{E}}$
(P2) $\mathbf{E} \xleftarrow{\mathcal{C}} \mathbf{B}$: ϵ / *fail*, t_1
(S4') $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{E}$: $\tilde{b}^{\mathbf{E}=\mathbf{B}}$
(P1) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $T^{\mathbf{A}}$
(P2') $\mathbf{A} \xleftarrow{\mathcal{C}} \mathbf{E}$: ϵ / *fail*, t_1
(P3) $\mathbf{A} \xrightarrow{\mathcal{C}} \mathbf{E}$: $P^{\mathbf{A}}, t_2$ / —
(P3') $\mathbf{E} \xrightarrow{\mathcal{C}} \mathbf{B}$: $P^{\mathbf{E}}, t_2$ / —

2. *Protocol and attack actions.*

(Sa) **A** performs step (Sa) of the protocol (prepares $\rho^{\mathbf{A}}$ and sends it in (S1)).
(Sc') **E** intercepts (S1) from **A** and stores $\rho^{\mathbf{A}}$ in her quantum memory. **E** sends an arbitrary number $n^{\mathbf{E}}$ (S2') to **A** to trigger **A**'s next message.
(Sd) **A** performs step (Sd) of the protocol: she sends (S3), i.e. $b^{\mathbf{A}}$.
(Sd') **E** intercepts (S3), measures $\rho^{\mathbf{A}}$ in **A**'s preparation bases $b^{\mathbf{A}}$, and obtains **A**'s rawkey $d^{\mathbf{A}}$.
(Sa') Using $d^{\mathbf{A}}$ and $b^{\mathbf{A}}$, **E** prepares an identical copy of $\rho^{\mathbf{A}}$ and sends it (S1') to **B**.
(Sb) , (Sc), (Sd"), (Se), (Sf') **E** (instead of **A**) and **B** follow the protocol–whereby sending (S2), (S3'), (S4)–until they obtain their sifted keys $s^{\mathbf{E}} \approx s^{\mathbf{B}}$.
(Pa') , (Pb) **E** (instead of **A**) and **B** follow the protocol–whereby sending (P1'),(P2)– and reconcile their sifted keys.

On receiving (P2) **E** has learned $M^{\mathbf{B}}$ and the tag t_1 and can now continue her communication with **A**.

(Se') **E** calculates a message $\tilde{b}^{\mathbf{E}=\mathbf{B}}$ such that (i) it is close to $b^{\mathbf{E}=\mathbf{B}}$ and (ii) $M^{\mathbf{A}\leftarrow\mathbf{E}} := (n^{\mathbf{E}}, \tilde{b}^{\mathbf{E}=\mathbf{B}}, \epsilon/\mathit{fail})$ collides with $M^{\mathbf{B}}$ under the inner hash function f, i.e. $f(M^{\mathbf{A}\leftarrow\mathbf{E}}) = f(M^{\mathbf{B}})$. **E** sends $\tilde{b}^{\mathbf{E}=\mathbf{B}}$ to **A** (S4').
(Sf) , (Pa) **A** calculates her sifted key $s^{\mathbf{A}}$, and sends (P1).
(Pb') **E** intercepts (P1) and can correct small errors introduced during quantum storage or measurement of $\rho^{\mathbf{A}}$. Using the original tag t_1, **E** forwards (P2')=(P2) to **A**.

(Pc) Since $f(M^{\mathbf{A}\leftarrow\mathbf{E}}) = f(M^{\mathbf{B}})$, **A** accepts the message as authentic, calculates $P^{\mathbf{A}}$ and $K^{\mathbf{A}} = P^{\mathbf{A}}(s^{\mathbf{A}})$, and sends (P3) with tag t_2.

(Pc') **E** calculates a PA function $P^{\mathbf{E}}$ (and obtains $K^{\mathbf{E}} = P^{\mathbf{E}}(\hat{s}^{\mathbf{B}})$) such that (i) $P^{\mathbf{E}}(\hat{s}^{\mathbf{B}}) = K^{\mathbf{A}}$, and (ii) $M^{\mathbf{E}\rightarrow\mathbf{B}} := (b^{\mathbf{E}}, T^{\mathbf{E}}, P^{\mathbf{E}})$ collides with $M^{\mathbf{A}}$ under f. **E** calculates $P^{\mathbf{E}}(\hat{s}^{\mathbf{B}})$, and sends (P3') with tag t_2 to **B**.

(Pd) **B** calculates $K^{\mathbf{B}} = P^{\mathbf{E}}(\hat{s}^{\mathbf{B}})$.

Eve shares a common "secret" key with Alice and Bob. In case that **E** cannot achieve condition (i) in step (Pc') she will get two individual keys with **A** and **B**. In both cases, protocol 2 is completely broken by the presented attack.

3.9. Discussion of attacks

The degree of success of the eavesdropper varies from protocol to protocol and ranges from a complete three party identity of the generated key – $K^{\mathbf{A}} = K^{\mathbf{E}} = K^{\mathbf{B}}$, to "separate worlds" outcome – $K^{\mathbf{A}} = K^{\mathbf{E}\leftrightarrow\mathbf{A}} \neq K^{\mathbf{E}\leftrightarrow\mathbf{B}} = K^{\mathbf{B}}$ (e.g. in a case of privacy amplification with no communication), to a successful attack over one of the legitimate parties (calling for a subsequent isolation of the other)– i.e. $K^{\mathbf{A}} \neq K^{\mathbf{E}} = K^{\mathbf{B}}$. Moreover the success can be achieved either deterministically or sometimes only probabilistically as in certain cases of encrypted transmission of error correction information.

This analysis underlines what was already mentioned in Section 3.2. As the attack mechanism fundamentally requires finding hash collisions of the internal authentication function that are *useful* to the eavesdropper, the different protocol versions discussed above, allow inequivalent optimal adversarial approaches. As it is to be expected, the availability of quantum resources simplifies the task of the eavesdropper but does not automatically lead to more powerful attacks. On the other hand immediate authentication also provides a leverage to the attacker as she does not have to correlate all her actions across the post-processing chain. This gives the somewhat counter-intuitive observation, that *fewer authentication tags* result in more difficulty for the attacker if he wants to keep the original message order! Furthermore sifting initiated by Bob also poses more difficulties to Eve as she can not learn the full information of Alice as is in the opposite case. Finally if part of the postprocessing information remains unknown to the eavesdropper, as in the case of encrypted reconciliation, then a deterministically successful attack strategy is not always guaranteed.

With all this said it must be underlined that Eve can find *useful collisions* only if she can fake the protocol communication by hiding her modifications in the typically available random degrees of freedom. If such are unavailable or strongly reduced (as e.g. in the case of protocols with delayed authentication or with communication-less privacy amplification) the room for attack is narrowed resulting in a number of cases in "separate world" or even "one-sided" adversarial success. Still in all discussed cases there always exists an attack strategy that renders the corresponding protocol version insecure.

4. Countermeasures

We will now propose a countermeasure that mitigates or, at a cost, prohibits the attacks exemplified in the previous section. One could consider encrypting parts of the communication between Alice and Bob [5, 21], but we will concentrate on strengthening the two-step authentication below. As we shall see, there are a number of possibilities ranging from increasing Eve's need for large computational power, all the way to information-theoretic security. As can be expected, the cost of this security improvement comes in the form of an increased secret key consumption.

Let us first consider the main enabler of the attacks presented in the previous section. The reason that the attacks are possible is that when Eve receives (or intercepts) Alice's message, she can immediately check if her message $m^{\mathbf{E}}$ coincides with Alice's under the publicly known hash function f. If not, Eve is free to choose another message $\tilde{m}^{\mathbf{E}}$ that does coincide with Alice's under f, although in some situations there is a small price to pay as described above. To prohibit this we should make it difficult or impossible for Eve to check for this coincidence. The essence of our proposed countermeasure is to use an extra bitsequence to make the output of the public hash function difficult to predict, or even secret, to Eve. This is done in the following way: prepend an extra bitsequence S to the message and authenticate the result. Instead of using the tag $t = g_K(m) = h_K(f(m))$, use the tag $t = g_K(S||m) = h_K(f(S||m))$. If, for example, S is random and secret to Eve, then the output $f(S||m)$ will also be secret to Eve, and she will not be able to search for coincidences in the above manner.

It should be stressed that S should be prepended to the message before applying f. The bitsequence S should *not* be concatenated with $f(m)$. The reason for this is fairly obvious. If S is concatenated with $f(m)$ so that $t = h_K(S||f(m))$ or $t = h_K(f(m)||S)$, then Eve can still apply her original attack strategy. All Eve needs in this case is still to find a message that collides with Alice's message under f. We should also stress that for certain classes of hash functions, prepending S to the message has advantages over appending to m (so that $t = h_K(f(m||S))$). When using iterative hash functions like **SHA-1** to calculate $f(m||S)$, Eve can ignore S and search instead for a message m' such that $f(m') = f(m)$. This is known as a partial-message collision attack, see Chapter 5 in Ref. [22]. If f is computed iteratively, $f(m') = f(m)$ will automatically give $f(m'||S) = f(m||S)$ (with appropriate block lengths). This is prohibited by prepending S instead, to use $f(S||m)$.

Of course, a random secret S would consume secret key, and this may not be desirable. Selecting S can be done in a few ways, and these are the alternatives (including a random secret S):

A salt, a random but fixed public bitstring, per device or per link. This would not make Eve's task much harder, but it would help a little in certain situations: for some messages, such as preparation and/or measurement settings, Eve does not need to use a random bitstring. She can use a fixed (random-looking) bitstring and for that message, a pre-calculated table of messages with low Hamming distance and

their corresponding intermediate tags [5]. Even though a full table might have an excessive number of entries (2^{256} is a large number), a partial table could ease Eve's calculational load (as in a rainbow table), or alternatively increase her probability of success. A salt would force Eve to create the table anew for each device or link.

A nonce, a random public bitstring, per authentication attempt. This may seem like a big improvement because it seems Eve cannot use a pre-calculated table, forcing her to make the calculations online. However, the nonce needs to be transmitted from Alice to Bob or vice versa, and is not separately authenticated, since this would need secret key better used elsewhere. A nonce can therefore be changed in transit by Eve, and this increases her possibilities. Authenticating a message from Alice to Bob, there are two sub-alternatives:

a) The nonce is generated by Alice and sent to Bob together with the tag, and Eve can change it in transit.
b) The nonce is generated by Bob and sent to Alice after he has received the message. One alternative for Eve is to commit to a message so that she can receive the nonce from Bob, and then change the nonce in transit. In effect, she can now change Alice's message since that contains the nonce.

In both cases, Eve needs to find a collision online, but Eve now has a message part that she can change to any value she desires. Therefore, her attack is easier in this setup, not more difficult.

Fixed secret key, a random but fixed secret bitstring, per device or per link. In this case, Eve cannot apply the previous attack on the authentication, because she cannot check for collisions directly since $f(S||m)$ is secret to her. To search for a message m' useful to Eve (i.e., having low Hamming distance to $m^{\mathbf{E}}$) such that $f(S||m') = f(S||m)$ has maximal probability (given the distribution of S) is computationally very costly. Moreover, we expect this maximal probability to be very low, but an upper bound is difficult to obtain and depends on details of the hash function, see below.

As regards using a fixed secret [23], if Eve has partial knowledge, no matter how small, on the secret key K identifying h_K, this information will accumulate over the rounds as information on S. Remember that after the initial pre-shared key is used up, K will consist of QKD-generated key that is ϵ-perfect (the trace distance between the probability distribution of the key and the uniform distribution is ϵ), where ϵ is nonzero. Therefore, after a large number of rounds, this reduces to using a random fixed public bitstring (salt) as discussed above.

Secret key, a random secret bitstring, per authentication attempt. Here also, Eve cannot apply the previous attack on the authentication, because she cannot check for collisions directly since $f(S||m)$ is secret to her. The situation is almost identical to the fixed secret key case but Eve's task is even harder as she cannot accumulate information on S.

This countermeasure is simple to implement, and the last alternative above seems

preferable, if only the key consumption is low. Choosing S to be of the same size as the tag gives a high computational load on Eve, and is efficient in terms of key consumption. It is, however, difficult to estimate the probability of success for Eve, if she has large computational power.

Let us examine what conditions need to be fulfilled to make the two-step authentication ITS. If the last alternative above is used, it is clear that we want a low probability of collision for a random value of S. And this is obtained if two distinct messages collide under f only for a small number of values of S. More formally, let $\mathcal{S}$ be the set of values of S. Then, if for any two distinct m_1, $m_2 \in \mathcal{M}$ $|\{S \in \mathcal{S} : f(S||m_1) = f(S||m_2)\}| \leq \epsilon'|\mathcal{S}|$, we automatically have a low collision probability. A close look at the above condition would tell us that it is precisely the condition for a family of hash functions indexed by S to be ϵ'-AU_2 (see Appendix C). The following theorem states that this condition is necessary and sufficient to restore ITS.

Theorem 1. *Let $\mathcal{M}$, $\mathcal{Z}$ and $\mathcal{T}$ be finite sets. Let $\mathcal{F}$ be a family of hash functions from $\mathcal{M}$ to $\mathcal{Z}$, $\mathcal{H}$ a family of SU_2 hash functions from $\mathcal{Z}$ to $\mathcal{T}$, and $\mathcal{G} := \mathcal{H} \circ \mathcal{F}$, where $\circ$ stands for element-wise composition. Then $\mathcal{G}$ is ϵ-ASU_2 if and only if $\mathcal{F}$ is ϵ'-AU_2, where $\epsilon = \epsilon'(1 - 1/|\mathcal{T}|) + 1/|\mathcal{T}|$.*

The proof can be found in Appendix C. Thus, to make the two-step authentication ITS, we should construct our fixed public hash function f with the help of an AU_2 hash function family $\mathcal{F}$ as follows:

$$f(S||m) = f_S(m), \ f_S \in \mathcal{F}. \tag{2}$$

In words, f separates S from the concatenation $S||m$ and uses it as index to select from the hash function family $\mathcal{F}$ an individual member f_S which is applied to the original message m.

Theorem 1 makes it possible to relate the message length $\log|\mathcal{M}|$, the security parameter ϵ', and the key consumption of the system. Let us aim for a final ϵ-ASU_2 family with $\epsilon = 2/|\mathcal{T}| - 1/|\mathcal{T}|^2$, i.e., $\epsilon' = 1/|\mathcal{T}|$. Then, the bound by Nguyen and Roscoe [24] is tight:

$$|\mathcal{F}| > |\mathcal{T}|\lceil \log|\mathcal{M}|/\log|\mathcal{Z}| - 1\rceil. \tag{3}$$

In [24] there are two lower bounds, but both can be written in this way. The bound applies when $\epsilon'|\mathcal{Z}| > 1 + \log|\mathcal{Z}|/(\log|\mathcal{M}| - \log|\mathcal{Z}|)$. The optimal family [24] is that of polynomial evaluation hashing over finite fields [25–27]. Therefore, using polynomial hashing with $|\mathcal{F}| = |\mathcal{Z}|$, we can authenticate messages as long as

$$\log|\mathcal{M}| < \Big(\frac{|\mathcal{Z}|}{|\mathcal{T}|} + 1\Big)\log|\mathcal{Z}|. \tag{4}$$

For example, if $|\mathcal{Z}| = 2^{256}$, $|\mathcal{T}| = 2^{64}$ and $\epsilon = 2^{-63} - 2^{-128}$, then messages of length $\log|\mathcal{M}| < 2^{200} \approx 10^{60}$ bits can be authenticated. The second step of the authentication

uses an SU_2 hash function $\mathcal{Z} \to \mathcal{T}$, which needs a key of length at least $\log|\mathcal{Z}| + \log|\mathcal{T}|$ bits [10, 28]. Thus, the total required key length is $2\log|\mathcal{Z}| + \log|\mathcal{T}|$, in this case 576 bits. By adjusting $|\mathcal{Z}|$ to the maximum message length, this scheme can authenticate one terabit (petabit, exabit, zettabit, yottabit) of data using 260 (280, 298, 318, 338) bits of secret key.

This construction makes the two-stage authentication ITS at the price of increasing the key consumption slightly. There are other efficient constructions of ASU_2 hash functions as well, see e.g., [9, 28–30]. Some of these authenticate message of arbitrary length with fixed key consumption at the price of a varying ϵ, while others have fixed ϵ but varying key consumption. They also vary in terms of their computational speed. The numbers are in the same range as the above presented ITS authentication, and all mentioned schemes are reasonably efficient.

5. Conclusions

The main conclusion of our extensive analysis is: *do not use non-ITS authentication in QKD if you want to achieve ITS security.* This may sound rather obvious but nevertheless in our oppinion it is always good to know what exactly goes wrong if you break the rules.

So, we have presented a comprehensive case study of attacks that compromise QKD in the non-ITS authentication setting of [3], that creates message tags by composing an inner public hash function with an outer function from a strongly universal hashing (SU_2) family. From the point of view of the attacker, who is equipped with unbounded computing resources, this composition has the following properties: (i) inserting a randomly chosen message or substituting messages with a randomly chosen message is as hard as in the SU_2 case and thus cannot be used in attacks, (ii) but more interestingly, substituting a message with another that collides under the public hash function will always work. As has been shown previously [3] property (i) does prohibit straightforward MITM attacks (cf. Definition 1).

The sophisticated MITM attacks dicussed here capitalize on property (ii) to successfully target many QKD protocol versions: protocols that use individual authentication of each message, or that use delayed authentication of all messages, protocols where Bob sends an acknowledgement message to trigger Alice's sifting message (containing her bases choice), or where Bob directly sends his bases choice, see Tables 2 and 3. All the attacks are enabled by the fact that the number of messages that collide with a given protocol message (or sequence of messages) of typically at least several hundred bits is extremely huge. Therefore, almost certainly (see Sec. 3.1) there exists at least one colliding message that allows the eavesdropper to perform her attack. In some attacks Eve needs less computing resources if she possesses quantum memory.

We stress that the discussed attack pattern is not restricted to one single instance, the specific authentication mechanism of Ref. [3] that we study here. We conjecture, that whenever property (ii) holds, i.e. collisions can be found, and the protocol does

not use additional secret key [5, 6] (e.g. for encryption of messages) the adversary can compromise the security of the key generated by QKD, following an interleaving approach along the lines of that discussed in this paper.

The countermeasures discussed in this paper use more secret key, specifically to prevent finding collisions. Prepending secret key material to the message, before applying the public hash function, will increase the computational resources needed for a successful attack substantially, at a low cost in terms of key material.

Furthermore, we can achieve Universally-Composable Information-Theoretic Security of the authentication scheme of [3] by replacing the publicly known hash function with an Almost Universal$_2$ function family. This requirement is necessary and sufficient for ITS of the two step authentication; the necessity of this condition is also a new result of this paper.

Acknowledgments

This work has been supported by the Vienna Science and Technology Fund (WWTF) via project ICT10-067 (HiPANQ) and also partly by the Austrian Research Promotion Agency (FFG) within the project Archistar (Bridge-2364544). C.P. would like to thank Shahram Mossayebi for reading a previous version of this paper and providing him with comments.

Appendix A. Proof of Lemma 1

Lemma 1. *Let $\mathcal{B}$ be the closed ball of all messages m having a Hamming distance to $m^{\boldsymbol{E}}$ not exceeding w:*

$$\mathcal{B} = \left\{ m : d_H(m, m^{\boldsymbol{E}}) \leq w \right\},$$

and let us assume that f maps all messages in $\mathcal{B}$ randomly onto $\mathcal{Z}$. Then the probability that at least one of the messages in $\mathcal{B}$ is validated by the given tag $t = h_K(f(m^{\boldsymbol{A}}))$ is

$$\mathcal{P}_{coll}^{succ} = \Pr\left\{ \exists \tilde{m}^{\boldsymbol{E}} \in \mathcal{B} : h_K(f(\tilde{m}^{\boldsymbol{E}})) = t) \right\} > 1 - \exp\left(-|\mathcal{B}||\mathcal{Z}|^{-1} \right).$$

For simplicity we can loosen the bound and replace $|\mathcal{B}|$ by $\binom{\ell}{w} < |\mathcal{B}|$, where ℓ is the length of the binary message $m^{\boldsymbol{E}}$.

Proof. By assumption, the probability that f maps any (randomly chosen) message m of $\mathcal{B}$ onto any fixed value z of $\mathcal{Z}$ is $1/|\mathcal{Z}|$:

$$m \in_R \mathcal{B}, \forall z \in \mathcal{Z} : \Pr\left\{ f(m) = z \right\} = 1/|\mathcal{Z}|. \tag{A.1}$$

Applying h_K to $f(m)$ and z in the argument of Pr (which potentially increases the value of the probability), setting $z = f(m^{\mathbf{A}})$, and using $t = h_K(f(m^{\mathbf{A}}))$ we obtain

$$m \in_R \mathcal{B} : \Pr\left\{ h_K(f(m)) = t \right\} \geq 1/|\mathcal{Z}|. \tag{A.2}$$

Consequently, the probability that t authenticates at least one message of all $|\mathcal{B}|$ different messages in $\mathcal{B}$ is at least $1-(1-|\mathcal{Z}|^{-1})^{|\mathcal{B}|}$, and using that $(1-1/n)^n < e^{-1}$ for $n>1$ finishes the proof. Finally, $|\mathcal{B}| = \sum_{k=0}^{w}\binom{\ell}{k} > \binom{\ell}{w}$. □

If desired, $1/|\mathcal{Z}|$ can be replaced by any lower bound on the probability to allow for non-uniform distributions.

Appendix B. Subsequence problem

Eve is given two fixed bit sequences, $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ (sifted key that Eve wants to achieve) and $d^{\mathbf{A}}$ (the raw key of Alice). Her goal is to find a *subsequence* of $d^{\mathbf{A}}$ that coincides with $s^{\mathbf{E}}$.

Appendix B.1. Algorithm that finds a subsequence

First we give a simple algorithm that takes two sequences $s = s_1|s_2|\ldots|s_m$, $S = S_1|S_2|\ldots|S_n$ as inputs and returns the index set $\mathcal{J} = \{j_1,\ldots,j_m\} = \{j_i : S_{j_i} = s_i\}$ if s is a subsequence of S (denoted $s \preccurlyeq S$).

Algorithm A find subsequence(s, S)
Input: two non-empty binary sequences s and S.
Output: index set $\mathcal{J}$ if s is a subsequence of S, else $\emptyset$.

$i \leftarrow 1$, $j \leftarrow 1$, $m \leftarrow$ length(s), $n \leftarrow$ length(S), $\mathcal{J} \leftarrow \emptyset$
do
  if $s_i = S_j$ **then** // we found one bit of s
    $\mathcal{J} \leftarrow \mathcal{J} \cup \{j\}$ // store position
    $i \leftarrow i+1$ // compare next bit of s
  endif
  $j \leftarrow j+1$ // compare next bit of S
while ($i \le m$ and $j \le n$) // neither end of s nor end of S reached
if $i \le m$ **then return** $\emptyset$ **endif** // end of s not reached, but end of S reached
return $\mathcal{J}$

Appendix B.2. Probability for finding a subsequence in a random sequence

We assume that both sequences consist of i.i.d. Bernoulli trials with $p(0) = p(1) = 1/2$ and calculate the (success) probability that $s \preccurlyeq S$.

$s \preccurlyeq S$ iff S is of the form

$$S = \bar{s}_1|\ldots|\bar{s}_1|\boldsymbol{s_1}|\bar{s}_2|\ldots|\bar{s}_2|\boldsymbol{s_2}|\ldots|\bar{s}_m|\ldots|\bar{s}_m|\boldsymbol{s_m}|x_1|x_2|\ldots. \tag{B.1}$$

Here, $\bar{s}_j$ denotes the negation of s_j (written above as $\boldsymbol{s_j}$ to improve readability), while each x_i can independently take value 0 or 1. All sequences $\bar{s}_j|\ldots|\bar{s}_j$ are optional. Let S be the *number of different* valid sequences, i.e. sequences S, that contain s as a

subsequence. Obviously S does not depend on s, but only on m and n. To calculate S we can therefore choose s to be the all zero sequence of length m. Consequently, S is equal to the number of different binary sequences of length n that contain at least m zeroes. The success probability

$$\mathrm{Prob}\{s \preccurlyeq S\} = \mathsf{S}/2^n = 2^{-n} \sum_{l=m}^{n} \binom{n}{l}. \tag{B.2}$$

Appendix B.3. Application to Eve's attack

Note that Eve wants to find the sifted key $s^{\mathbf{E}\leftrightarrow\mathbf{B}}$ in Alice's raw key $d^{\mathbf{A}}$. If both bases are used with equal probability (as in standard symmetric BB84), then $m \approx n/2$. Obviously,

$$\mathrm{Prob}\{s \preccurlyeq S\} > \frac{1}{2} \Leftarrow m \leq \lfloor n/2 \rfloor. \tag{B.3}$$

However, it is not necessary, that s is an *exact* subsequence of S. We can allow for some errors that will be removed during the subsequent error correction step. Using Hoeffding's inequality (Theorem 1 in Ref. [31]) we can give a non-tight (but exponential) lower bound on $\mathrm{Prob}\{\tilde{s} \preccurlyeq S\}$ if we allow for approximately k errors in the resulting subsequence $\tilde{s}$:

$$\mathrm{Prob}\{\tilde{s} \preccurlyeq S\} \geq 1 - \exp\left(-\frac{2k^2}{n}\right) \Leftarrow \tilde{m} = \lfloor n/2 \rfloor - k. \tag{B.4}$$

Here, only $\tilde{m}$ bits of s form a subsequence of S. For moderate values of k this probability reaches almost unity.

Appendix C. Universal hash functions and proof of Theorem 1

In the following we give definitions of (Almost) Universal$_2$ and Strongly Universal$_2$ hash function families; see e.g., [10].

Definition 3 (ϵ-Almost Universal$_2$ (ϵ-AU$_2$) hash functions)**.** *Let $\mathcal{M}$ and $\mathcal{T}$ be finite sets. A class $\mathcal{H}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is* ϵ-Almost Universal$_2$ *if there exist at most $\epsilon|\mathcal{H}|$ hash functions $h \in \mathcal{H}$ such that $h(m_1) = h(m_2)$ for any two distinct $m_1, m_2 \in \mathcal{M}$.*

If $\epsilon = 1/|\mathcal{T}|$, then $\mathcal{H}$ is called Universal$_2$ (U$_2$)*.*

Definition 4 (ϵ-Almost Strongly Universal$_2$ (ϵ-ASU$_2$) hash functions)**.** *Let $\mathcal{M}$ and $\mathcal{T}$ be finite sets. A class $\mathcal{H}$ of hash functions from $\mathcal{M}$ to $\mathcal{T}$ is ϵ-ASU$_2$ if the following two conditions are satisfied:*

(a) The number of hash functions in $\mathcal{H}$ that takes an arbitrary $m_1 \in \mathcal{M}$ to an arbitrary $t_1 \in \mathcal{T}$ is exactly $|\mathcal{H}|/|\mathcal{T}|$.

(b) The fraction of those functions that also takes an arbitrary $m_2 \neq m_1$ in $\mathcal{M}$ to an arbitrary $t_2 \in \mathcal{T}$ (possibly equal to t_1) is at most ϵ.

If $\epsilon = 1/|\mathcal{T}|$, then $\mathcal{H}$ is called Strongly Universal$_2$ (SU$_2$)*.*

Below, we have restated Theorem 1 together with its proof. This theorem states that the composition of a hash function family with an SU_2 family will form an ASU_2 family if and only if the first family in the composition is AU_2. The "if" part follows from the composition theorem [10], but the below proof simultaneously handles "if and only if".

Theorem 1. *Let $\mathcal{M}$, $\mathcal{Z}$ and $\mathcal{T}$ be finite sets. Let $\mathcal{F}$ be a family of hash functions from $\mathcal{M}$ to $\mathcal{Z}$, $\mathcal{H}$ a family of SU_2 hash functions from $\mathcal{Z}$ to $\mathcal{T}$, and $\mathcal{G} := \mathcal{H} \circ \mathcal{F}$, where $\circ$ stands for element-wise composition. Then $\mathcal{G}$ is ϵ-ASU_2 if and only if $\mathcal{F}$ is ϵ'-AU_2, where $\epsilon = \epsilon'(1 - 1/|\mathcal{T}|) + 1/|\mathcal{T}|$.*

Proof. For $\mathcal{G}$ to be ϵ-ASU_2, there are two requirements (Definition 4). The first, on $|\{g : g(m) = t\}|$, needs no properties of $\mathcal{F}$, because, for any $m \in \mathcal{M}$ and $t \in \mathcal{T}$,

$$\begin{aligned} |\{g : g(m) = t\}| &= \sum_z |\{f : f(m) = z\}||\{h : h(z) = t\}| \\ &= \sum_z |\{f : f(m) = z\}| \frac{|\mathcal{H}|}{|\mathcal{T}|} = |\mathcal{F}| \frac{|\mathcal{H}|}{|\mathcal{T}|} = \frac{|\mathcal{G}|}{|\mathcal{T}|}. \end{aligned} \tag{C.1}$$

The second requirement is a bound for

$$\begin{aligned} &|\{g : g(m_1) = t_1, g(m_2) = t_2\}| \\ &\quad = \sum_z |\{f : f(m_1) = f(m_2) = z\}||\{h : h(z) = t_1, h(z) = t_2\}| \\ &\qquad + \sum_{z_1 \neq z_2} |\{f : f(m_1) = z_1, f(m_2) = z_2\}||\{h : h(z_1) = t_1, h(z_2) = t_2\}|, \end{aligned} \tag{C.2}$$

for any two distinct m_1, $m_2 \in \mathcal{M}$. If $t_1 \neq t_2$, the first term above will be zero because $h(z)$ will never equal both t_1 and t_2. If instead $t_1 = t_2 = t$, the first term simplifies to

$$\sum_z |\{f : f(m_1) = f(m_2) = z\}||\{h : h(z) = t\}| = |\{f : f(m_1) = f(m_2)\}| \frac{|\mathcal{H}|}{|\mathcal{T}|}. \tag{C.3}$$

The second term is

$$\begin{aligned} &\sum_{z_1 \neq z_2} |\{f : f(m_1) = z_1, f(m_2) = z_2\}||\{h : h(z_1) = t_1, h(z_2) = t_2\}| \\ &\qquad = \big(|\mathcal{F}| - |\{f : f(m_1) = f(m_2)\}|\big) \frac{|\mathcal{H}|}{|\mathcal{T}|^2} \end{aligned} \tag{C.4}$$

and this can be bounded by $|\mathcal{G}|/|\mathcal{T}|^2$ only using properties of $\mathcal{H}$. Thus, if $t_1 = t_2$ the first term needs a bound for collisions within $\mathcal{F}$, while the second only needs properties of $\mathcal{H}$, and we obtain

$$|\{g : g(m_1) = t_1, g(m_2) = t_2\}| = |\{f : f(m_1) = f(m_2)\}| \Big(\delta_{t_1,t_2} - \frac{1}{|\mathcal{T}|}\Big) \frac{|\mathcal{H}|}{|\mathcal{T}|} + \frac{|\mathcal{G}|}{|\mathcal{T}|^2}, \tag{C.5}$$

where $\delta_{t_1,t_2} = 1$ if $t_1 = t_2$ and 0 otherwise. This makes the second requirement on $\mathcal{G}$ equivalent to $\mathcal{F}$ being ϵ'-AU_2:

$$\begin{aligned} |\{g : g(m_1) = t_1, g(m_2) = t_2\}| \leq \epsilon \frac{|\mathcal{G}|}{|\mathcal{T}|} = \epsilon' \Big(1 - \frac{1}{|\mathcal{T}|}\Big) \frac{|\mathcal{G}|}{|\mathcal{T}|} + \frac{|\mathcal{G}|}{|\mathcal{T}|^2} \\ \iff \\ |\{f : f(m_1) = f(m_2)\}| \leq \epsilon' |\mathcal{F}|. \end{aligned} \tag{C.6}$$

□

References

[1] Menezes A, van Oorschot P C and Vanstone S A 1996 *Handbook of Applied Cryptography* (CRC Press) ISBN 0-8493-8523-7

[2] Portmann C 2012 URL `http://arXiv.org/abs/1202.1229v2`

[3] Peev M, Nölle M, Maurhardt O, Lorünser T, Suda M, Poppe A, Ursin R, Fedrizzi A and Zeilinger A 2005 *International Journal of Quantum Information* **3** 225–231

[4] Beth T, Müller-Quade J and Steinwandt R 2005 *Quantum Information and Computation* **5** 181–186

[5] Abidin A and Larsson J Å 2009 *International Journal of Quantum Information* **7** 1047–1052

[6] Peev M, Pacher C, Lorünser T, Nölle M, Poppe A, Maurhart O, Suda M, Fedrizzi A, Ursin R and Zeilinger A 2009 *International Journal of Quantum Information* **7** 1401–1407

[7] Wegman M N and Carter J L 1981 *J. Comput. Syst. Sci.* **22** 265

[8] Carter J L and Wegman M N 1979 *J. Comput. Syst. Sci.* **18** 143

[9] Krawczyk H 1994 Lfsr-based hashing and authentication *CRYPTO '94* (*Lecture Notes in Computer Science* vol 839) ed Desmedt Y (Springer 1994) pp 129–139

[10] Stinson D R 1991 Universal hashing and authentication codes *CRYPTO '91* (*Lecture Notes in Computer Science* vol 576) ed Feigenbaum J (Springer 1992) pp 74–85

[11] Mehlhorn K and Vishkin U 1984 *Acta Inf.* **21** 339–374

[12] Shoup V 1996 On fast and provably secure message authentication based on universal hashing *CRYPTO '96* (*Lecture Notes in Computer Science* vol 1109) ed Koblitz N (Springer 1996) pp 313–328

[13] Bennett C H, Bessette F, Brassard G, Salvail L and Smolin J A 1992 *J. Cryptology* **5** 3–28

[14] Scarani V, Bechmann-Pasquinucci H, Cerf N J, Dušek M, Lütkenhaus N and Peev M 2009 *Rev. Mod. Phys* **81** 1301–1350

[15] Ben-Or M and Mayers D 2004 URL `http://arXiv.org/abs/quant-ph/0409062`

[16] Ben-Or M and Mayers D 2005 The Universal Composable Security of Quantum Key Distribution *Proceedings of TCC 2005, Cambridge, MA, USA* (*Lecture Notes in Computer Science* vol 3378) ed Kilian J (Springer 2005) pp 386–406 URL `http://arXiv.org/abs/quant-ph/0409078`

[17] Renner R and König R 2005 Universally composable privacy amplification against quantum adversaries *Proceedings of TCC 2005, Cambridge, MA, USA* (*Lecture Notes in Computer Science* vol 3378) ed Kilian J (Springer 2005) pp 407–425

[18] Müller-Quade J and Renner R 2009 *New Journal of Physics* **11** 085006

[19] Sasaki M, Fujiwara M, Ishizuka H, Klaus W, Wakui K, Takeoka M, Miki S, Yamashita T, Wang Z, Tanaka A, Yoshino K, Nambu Y, Takahashi S, Tajima A, Tomita A, Domeki T, Hasegawa T, Sakai Y, Kobayashi H, Asai T, Shimizu K, Tokura T, Tsurumaru T, Matsui M, Honjo T, Tamaki K, Takesue H, Tokura Y, Dynes J F, Dixon A R, Sharpe A W, Yuan Z L, Shields A J, Uchikoga S, Legré M, Robyr S, Trinkler P, Monat L, Page J B, Ribordy G, Poppe A, Allacher A, Maurhart O, Länger T, Peev M and Zeilinger A 2011 *Opt. Express* **19** 10387–10409

[20] Gilbert G and Hamrick M 2000 Practical Quantum Cryptography: A Comprehensive Analysis (Part One) URL `http://arXiv.org/abs/quant-ph/0009027v5`

[21] Lütkenhaus N 1999 *Phys. Rev. A* **59** 3301–3319

[22] Ferguson N, Schneier B and Kohno T 2010 *Cryptography Engineering* (Wily Publishing, Inc.)

[23] Abidin A and Larsson J Å 2011 Security of authentication with a fixed key in quantum key distribution URL `http://arXiv.org/abs/1109.5168v1`

[24] Nguyen L H and Roscoe A W 2010 New combinatorial bounds for universal hash functions Universal Computing

[25] den Boer B 1993 *J. Comp. Sec.* **2** 65–72

[26] Johansson T, Kabatianskii G and Smeets B 1993 On the relation between a-codes and codes correcting independent errors *Advances in Cryptology, EUROCRYPT 1993* (*Lecture Notes in Computer Science* vol 765) (Springer-Verlag) pp 1–11

[27] Taylor R 1994 An integrity check value algorithm for stream ciphers *Advances in Cryptology - CRYPTO '93* (*Lecture Notes in Computer Science* vol 773) ed Stinson D (Springer-Verlag) pp 40–48

[28] Bierbrauer J, Johansson T, Kabatianskii G and Smeets B 1994 On families of hash functions via geometric codes and concatenation *CRYPTO '93* (*Lecture Notes in Computer Science* vol 773) ed Stinson D (Springer-Verlag) pp 331–342

[29] Bierbrauer J 1997 *Designs, Codes and Cryptography* **11** 207–221

[30] Abidin A and Larsson J Å 2012 New universal hash functions *WEWoRC 2011* (*Lecture Notes in Computer Science* vol 7242) ed Lucks S and Armknecht F (Springer-Verlag) pp 99–108

[31] Hoeffding W 1963 *J. Amer. Statistical Assoc.* **58** 13–30